MECANISMOS DE OPTIMIZACIÓN EN EL PREPROCESADO PARA H.265/HEVC

FERNANDO SÁNCHEZ PASTOR

MÁSTER EN INVESTIGACIÓN EN INFORMÁTICA, FACULTAD DE INFORMÁTICA,
UNIVERSIDAD COMPLUTENSE DE MADRID

Trabajo Fin Máster en Ingeniería de Computadores

Septiembre de 2016
Calificación: 8.2

Directores:

Guillermo Botella Juan/ Alberto A. del Barrio García

Autorización de Difusión

FERNANDO SÁNCHEZ PASTOR

05-09-2016

El abajo firmante, matriculado en el Máster en Investigación en Informática de la Facultad de Informática, autoriza a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el presente Trabajo Fin de Máster: "Mecanismos de optimización en el preprocesado para H.265/HEVC", realizado durante el curso académico 2016-2017 bajo la dirección de Guillermo Botella Juan y Alberto del Barrio en el Departamento de Arquitectura de Computadores y Automática, y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo.

Resumen en castellano

En esta memoria se ha implementado una etapa de preprocesado que sirva como primera fase en el proceso de codificación de vídeo. Esta etapa integra dos variedades del filtro de mediana (3×3 y 5×5) y un operador. Dicho operador lleva a cabo el cálculo del gradiente de los píxeles que conforman una imagen o fotograma con objeto de filtrar después aquellos que están por debajo de un determinado valor (threshold).

El cálculo de dicho threshold se realiza de manera empírica mediante dos procesos distintos. En el primero se obtienen valores de luminancia y crominancia de píxeles que integran bordes para encontrar aquel que tenga el valor mínimo, mientras que en el segundo se calcula la tasa de píxeles que forman parte de bordes.

Una vez se ha realizado el cálculo anterior, se han utilizado distintos valores de threshold, distintas variedades de filtro de mediana y distintos valores de QP (calidad) con objeto de parametrizar las codificaciones que hacen uso de esta nueva etapa. Posteriormente a dichas codificaciones, se han obtenido los tamaños de los bitstreams de salida y se ha evaluado la calidad de los vídeos decodificados o reconstruidos mediante dos métricas objetivas: PSNR y SSIM.

Las codificaciones que no utilizan etapa de preprocesado también han sido evaluadas mediante dichas métricas y comparadas con aquellas que sí integran dicha etapa. Los resultados obtenidos dejan patente el compromiso existente entre tamaño de bitstream y calidad, siendo más representativos los de la métrica SSIM, estando esta última más relacionada con la percepción de la imagen por parte del HVS (sistema visual humano). Como resultado, se obtiene para esta métrica tasas de compresión mayores que las alcanzadas sin preprocesamiento, con pérdidas de calidad prácticamente inapreciables.

Palabras clave

Bitstream, bitrate, frame rate, PSNR, SSIM, códec, HEVC, H.265, H.264, AVC, métrica, QP, filtro, operador, threshold, RGB, YUV, HVS.

Resumen en inglés

This MSc Thesis has implemented a preprocessing stage to work as a first step in the process of encoding videos. This stage integrates two varieties of median filter (3×3 and 5×5) and an operator. Such operator performs the calculation of the gradient of the pixels that compose an image or frame in order to filter those below a certain value (threshold).

The calculation of this threshold is empirically performed by two different processes. The first one, deals with the luminance and chrominance pixel values that belong to boundaries in order to find the one with the minimum value. On the other hand, the second process is based on the estimation of the pixel rate just considering edge areas.

Once the aforementioned calculation has been done, different threshold values, parameters as well as varieties of filters are applied to produce the video encoding. After performing the encoding, the bitstream sizes have been obtained and the quality of the decoded or reconstructed videos has been evaluated using two objective metrics: PSNR and SSIM.

The encodings that do not use a preprocessing stage have also been evaluated with these metrics and compared with those that do integrate this stage. The results clearly demonstrate the existence of a trade-off between size and quality bitstream, especially when considering the evaluation with the SSIM metric, as it is more related with the perception of the image by the Human View System. To conclude, the main contribution of this MSc Thesis is the achievement of higher compression rate with a negligible quality loss, in comparison with the conventional video coding processing scheme.

Keywords

Bitstream, bitrate, frame rate, PSNR, SSIM, códec, HEVC, H.265, H.264, AVC, metric, QP, filtro, operador, threshold, RGB, YUV, HVS.

A mis padres

A Victoria

A mi ahijado Martín

Índice de contenidos

Índice de tablas

Índice de figuras

Agradecimientos

Quiero expresar mi agradecimiento a Guillermo Botella y a Alberto del Barrio por su apoyo y comprensión a la hora de realizar este trabajo.

1. Introducción

En la actualidad se está incrementando cada vez más la calidad de los contenidos multimedia que se producen, particularmente los relativos a vídeo. Uno de los medios más habituales de transmisión de vídeos es internet, sin embargo, el ancho de banda con el que se cuenta para realizar dicha transmisión es limitado. Por ello será fundamental la labor desempeñada por los codificadores/decodificadores (a partir de aquí códecs). Éstos nos permitirán reducir el tamaño del vídeo original de manera que se pueda adecuar al ancho de banda sin comprometer la calidad. De forma general podemos considerar que los estándares de video parten desde el H.261 [1] y han evolucionado hasta el actual H.265 o también conocido como HEVC [2].

1.1 Objetivos

Con la finalidad de ayudar al códec a llevar a cabo su tarea, es posible tratar el vídeo antes de que éste sea codificado. A dicho tratamiento previo se le denomina preprocesado y tiene dos propósitos fundamentales: reducir el tamaño y mejorar la calidad del *bitstream* que el códec proporcionará como salida. Es decir, que al aplicar el mencionado preprocesado el códec ya no recibirá como entrada el vídeo original sino el vídeo obtenido al tratar el original.

Para comprobar si efectivamente la etapa de preprocesado logra su cometido, basta con medir los resultados de las salidas obtenidas cuando la entrada es el vídeo tratado y compararlas con las que se obtienen cuando la entrada es el vídeo original. Las mediciones se llevarán a cabo sobre los dos parámetros antes citados: tamaño de *bitstream* y calidad.

Se pueden llevar a cabo distintas modificaciones en el vídeo original a la hora de preprocesarlo, arrojando éstas resultados distintos para los dos factores que se desean optimizar. Será necesario por tanto llevar a cabo comparaciones entre los resultados de los distintos preprocesados para evaluar cuál es el más conveniente.

1.2 Factores y medición de la calidad

Dentro del proceso de codificación de los estándares de vídeo MPEG podemos apreciar dos técnicas de codificado*: intra coding* e *inter coding*. En ambas técnicas tendrá lugar una etapa de cuantización, que a partir del estándar H.263 [3] se puede parametrizar mediante el *quantization parameter* (QP) que da idea del nivel de detalle que el codificador aplicará a la hora de llevar a cabo su tarea.

A modo de ejemplo, en el códec H.265 (HEVC) los valores del QP pueden variar de 0 (no habrá perdida en la compresión, o lo que es lo mismo no se aplicará cuantización) a 51. Cuanto más pequeño es este valor, más nivel de detalle estaremos aplicando (y mayor tamaño del archivo *bitstream* de salida), como se aprecia en la Figura 1. Por tanto, cuanto mayor sea este parámetro más pérdida de calidad estaremos teniendo y por ende, estaremos introduciendo un mayor nivel de distorsión.

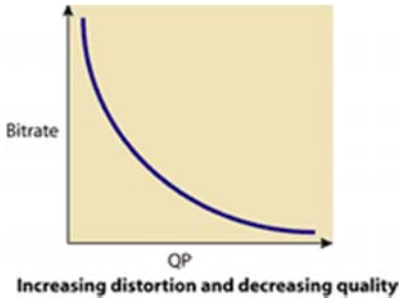

Figura 1: Comportamiento QP-Bitrate **[4]**.

HM16x es uno de las implementaciones software [5] que utilizan el estándar de compresión de vídeo HEVC o H.265. Realizando codificaciones, mediante dicho software de un vídeo de resolución QCIF (176x144) para distintos valores de QP constante se obtienen los valores de tamaño de *bitstream* expresados en bytes en la Tabla 1.

QP	TAMAÑO (bitstream)
QP15	4.368.113
QP20	3.037.536
QP25	1.967.309
QP30	1.144.132
QP35	624.453
QP40	316.756
QP45	135.495
QP50	51.149

Tabla 1. Relación entre QP y tamaño de bitstream.

Mediante este parámetro aplicamos la constricción de codificar fijando una determinada calidad, desconociendo el *bitrate* y el tamaño de *bitstream* resultante, frente a una codificación en la que se fije el *bitrate* en promedio y el códec obtenga como resultado el mejor *bitstream* posible para dicho *bitrate* tal y como se aprecia en la Tabla 2.

Bitrate	Tamaño (bitstream)
1Mbps	2.038.039
2Mbps	3.757.202
3Mbps	5.569.502

Tabla 2. Relación entre bitrate y tamaño de bitstream.

Para esta segunda alternativa el códec irá variando el QP a medida que vaya realizando la codificación, de manera que se alcance el *bitrate* utilizado como parámetro.

X265 [2] por otra parte, es otro software basado en licencia libre que implementa el estándar HEVC, este códec nos muestra el QP en promedio una vez ha finalizado la codificación en la que se utilizó el *bitrate* como parámetro.

Bitrate	QP
1Mbps	28.59
2Mbps	21.34
3Mbps	15.73

Tabla 3. Relación entre bitrate y QP.

Se puede calcular el *bitrate* de los *bitstreams* obtenidos mediante el primer códec a partir de su tamaño, su número de frames (300) y su *frame rate* (25 fps). Dividiendo el número de frames del *bitstream* entre su *frame rate* se obtiene la duración del vídeo (en este caso 12 segundos). Realizando una nueva división que tenga como dividendo el tamaño del *bitstream* y como divisor la duración del mismo se halla el *bitrate*. Se puede expresar lo anterior mediante la siguiente expresión: $bitrate = \frac{Tamaño\ bitstream * frame\ rate}{Número\ de\ frames}$. Aplicando la fórmula en el caso en que el QP es igual a 15 nos queda: $bitrate = \frac{4368113\ bytes * 25\ fps}{300\ frames}$ es decir 364.009 bytes/s o expresado en su unidad más habitual 2,77 Mbps. En la Tabla 4 se recogen este *bitrate* y los calculados para el resto de los QP.

QP	Bitrate
QP15	2,77 Mbps
QP20	1,93 Mbps
QP25	1,25 Mbps
QP30	0,73 Mbps
QP35	0,40 Mbps
QP40	0,20 Mbps
QP45	0,09 Mbps
QP50	0,03 Mbps

Tabla 4. Relación entre QP y bitrate.

Figura 2: Frames para QP's 15 (izq), 30 (medio) y 50 (der.).

Se puede observar el mismo *frame* para diferentes valores de QP, apreciándose un descenso de calidad del segundo *frame* con respecto al primero. Para valores muy altos de QP como puede ser el 50 (prácticamente el máximo), el descenso en calidad de la imagen se torna inaceptable como se puede ver en la Figura 2.

Figura 3. Frames para Bitrates 1Mbps (izq), 2 Mbps (centro) y 3Mbps (derecha).

La Figura 3 ilustra la parametrización con distintos *bitrates* sobre un mismo frame. Sin embargo las diferencias son prácticamente inapreciables. El *bitrate* está relacionado con la calidad, sin embargo podemos tener vídeos con *bitrates* diferentes donde el sistema visual humano (HVS) no aprecie diferencias entre uno y otro.

Para verificar calidades parecidas para los valores de QP igual a 15 y *bitrate* igual a 3 Mbps (Figura 4), así como para el valor de QP=30 y *bitrate* 2 Mbps (Figura 5), se puede utilizar una métrica de calidad objetiva como la PSNR. Para ello, se puede hacer uso de una herramienta de evaluación de calidad de vídeo como es *Intel Video Quality Caliper* [6].

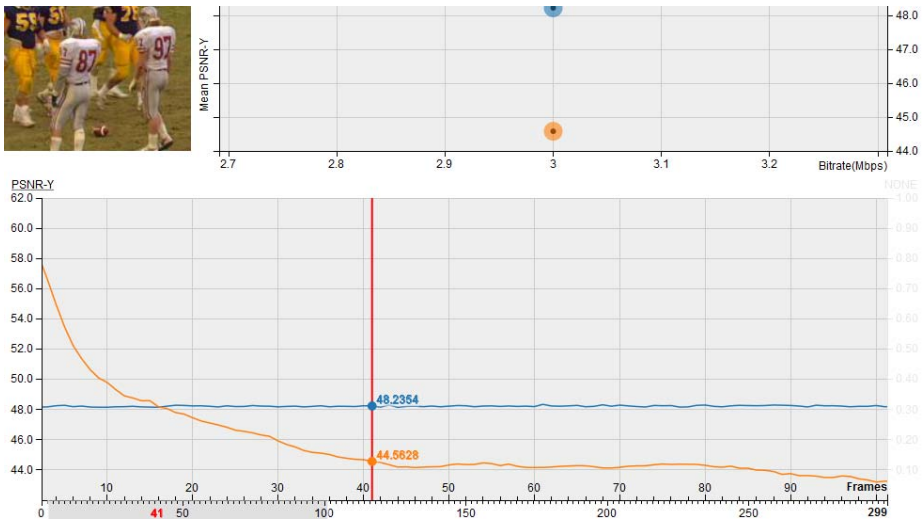

Figura 4. Visualización de la métrica PSNR en Intel Video Quality Caliper con QP =15.

Dicha herramienta recibirá como entrada el vídeo original y los dos archivos de reconstrucción (bitstream en el contenedor del archivo original) obtenidos como salida de los códecs HM16 (QP = 15) y x265 (*bitrate* = 3Mbps), siendo los valores obtenidos como PSNR media 48,22 (azul) y 44,58 (naranja) respectivamente tal y como se muestra en la Figura 4. En la Tabla 5 se muestra la relación obtenida entre QP y PSNR.

QP	PSNR
QP15	48,22
QP20	43,29
QP25	38,44
QP30	34,03
QP35	30,36
QP40	27,11
QP45	24,10
QP50	21,78

Tabla 5. Relación entre QP y PSNR.

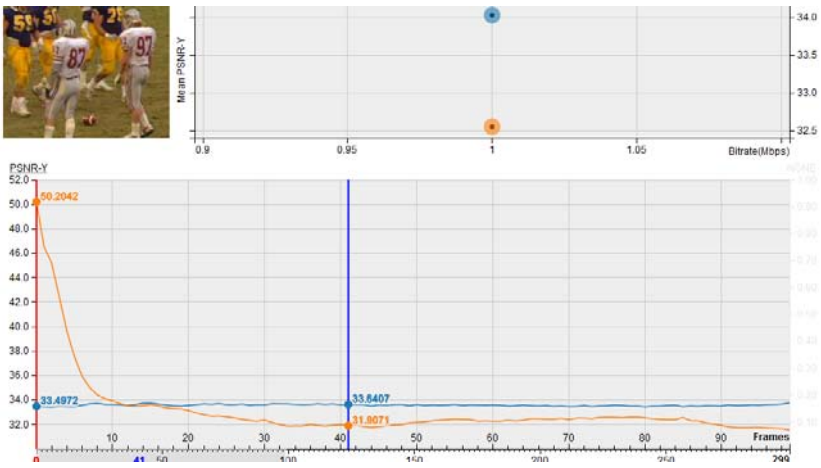

Figura 5. Visualización de la métrica PSNR en Intel Video Quality Caliper con QP=30.

Para QP igual a 30 y un *bitrate* de 1Mbps se obtienen también valores semejantes para dicha métrica, siendo la media de ésta 34,03 para el primero (QP = 30) y 32,55 para el segundo (*bitrate* = 1Mbps). En la Tabla 6 se muestra también la relación entre bitrate y PSNR.

Bitrate	PSNR
1Mbps	32,55
2Mbps	39,18
3Mbps	44,58

Tabla 6. Relación entre bitrate y PSNR.

Podemos por tanto fijar valores para sendos parámetros (QP y *bitrate*) en función de la calidad y el tamaño que queramos en el vídeo resultante. Se puede ver en la Figura 6 como la relación entre tamaño de *bitstream* y *bitrate* es directamente proporcional, mientras que la relación con el QP se torna inversamente proporcional.

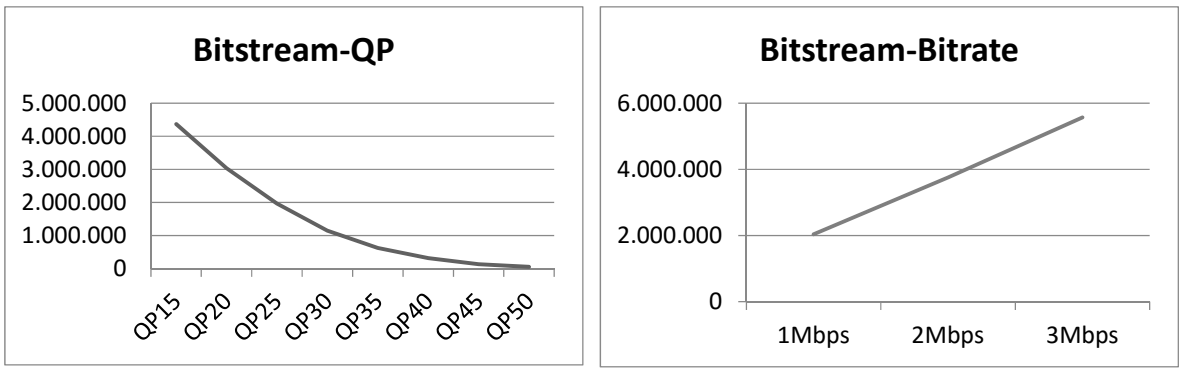

Figura 6. Relación Tamaño Bitstream-QP y Tamaño Bitstream-Bitrate.

La relación de proporcionalidad de QP y *bitrate* con calidad se da también, en este caso, de manera directa e inversa respectivamente, como se puede constatar en la Figura 7.

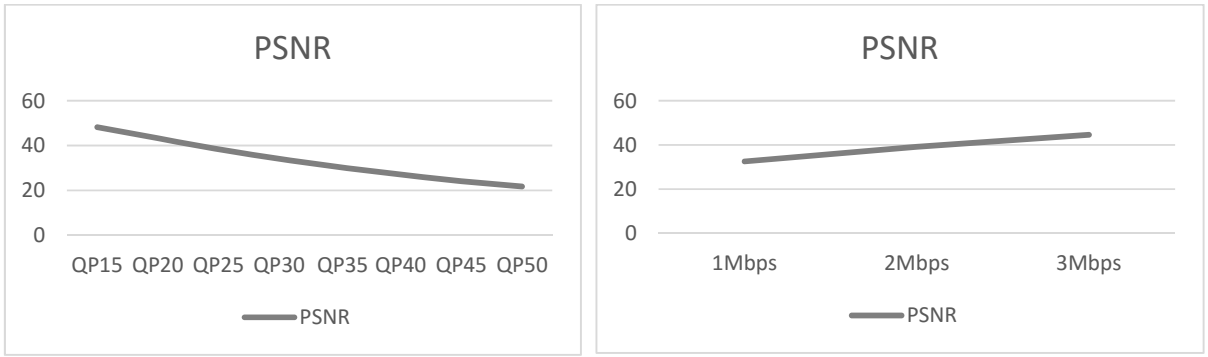

Figura 7. Relación PSNR-QP y PSNR-Bitrate.

Se han expuesto en esta sección las relaciones entre parámetros fundamentales en codificación de vídeo como bitstream, QP, etc., quedando patente el *trade-off* entre *bitrate* y tamaño de *bitstream*. Será interesante comprobar si añadiendo una etapa de preprocesado al codificador este problema puede ser mitigado.

2. Trabajo relacionado: Códecs de vídeo

En este apartado se abordará una introducción al fundamento de los codificadores de video H.264, HEVC y VP9.

2.1 H.264

El estándar H.264 fue publicado en el año 2003 y fue desarrollado conjuntamente por el International Telecommunication Union (ITU), en concreto por el Video Coding Experts Group (VCEG), que forma parte del sector de este organismo que coordina los estándares de telecomunicaciones y el ISO/IEC Moving Picture Experts Group (MPEG).

2.1.1 Predicción Intra-fotograma

Este tipo de predicción se lleva a cabo sin tener en cuenta ningún otro fotograma que no sea el actual. Al igual que en estándares previos, los macrobloques están formados por 16×16 muestras de luminancia y dos macrobloques de muestras de crominancia cuyas dimensiones dependerán del *chroma subsampling* del fotograma a codificar.

2.1.1.1 Predicción 4×4

El bloque 4×4 del que se realiza la predicción se calcula a partir de las muestras superiores (inicialmente del bloque 4×4 inmediatamente superior al actual), así como de los que se encuentran a su izquierda (inicialmente del bloque 4×4 inmediatamente a la izquierda del actual). Existe también una independencia dentro de cada slice (nuevo término que aparece en el estándar H.264 que hace referencia a segmentos con dimensiones iguales o superiores a las de un macrobloque e inferiores a las de un fotograma o *frame*) ya que sólo se utilizan para la predicción aquellas muestras incluidas dentro del mismo slice en el que se encuentra el bloque 4×4 a calcular [7] (ver Figura 8).

Dentro de este se dispone a su vez de 9 modos opcionales (0-8) para cada bloque 4×4:

Modo 0. La extrapolación del bloque 4×4 se realiza verticalmente, es decir, la muestra de luminancia a predecir toma el valor de la inmediatamente superior (todas las muestras requeridas deben estar disponibles)

Modo 1. La extrapolación del bloque 4×4 se realiza horizontalmente, es decir la muestra de luminancia a predecir toma el valor de la inmediatamente a la izquierda de ella (todas las muestras requeridas deben estar disponibles)

Modo 2 (DC). No todas las muestras requeridas deben estar disponibles. El bloque 4×4 se calculará haciendo la media entre las muestras inferiores del bloque inmediatamente superior y las muestras más a la derecha del bloque inmediatamente a la izquierda.

Modo 3. Se realiza una interpolación de la muestra actual con respecto a la inmediatamente superior a la derecha.

Modo 4. Se realiza una extrapolación de la muestra actual con respecto a la inmediatamente superior a la izquierda.

Modo 5. La extrapolación se realiza de manera similar a la anterior, pero en ángulo de 26,6 grados con la vertical en vez de 45.

Modo 6. La extrapolación se realiza de manera similar a la anterior, pero en ángulo de 26,6 grados por debajo de la horizontal.

Modo 7. La extrapolación se realiza de manera similar a la del modo 3, pero en ángulo de 26,6 grados con la vertical en vez de 45.

Modo 8. La extrapolación se realiza de manera similar a la del modo 6, pero en ángulo de 26,6 grados por encima de la horizontal. [8]

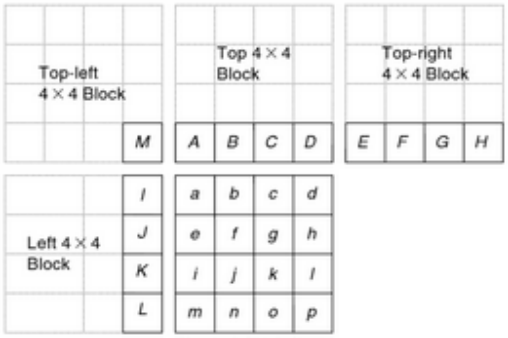

Figura 8. División en bloques para la predicción 4×4 **[9]**.

Para los modos del 3 al 8, las muestras a predecir se calculan mediante un promedio en el que se otorgan diferentes pesos a las muestras que se utilizan para dicha predicción.

2.1.1.2 Predicción 16×16

Modo 0. Verticalmente

Modo 1. Horizontalmente

Modo2 (DC). El cálculo de la media se realiza de la misma manera que en el 4×4 sólo que ahora en el *macrobloque* de 16×16.

Se aplica para la predicción una función lineal sobre las muestras inferiores del *macrobloque* superior y las muestras más a la derecha del macrobloque de la izquierda. Esta estrategia funciona de forma satisfactoria en áreas que experimentan una suave variación de la luminancia. [9]

2.1.1.3 Predicción 8×8

Tiene los mismos modos que en la predicción 16×16, con similar comportamiento, salvo por las dimensiones del bloque que en este caso son 8×8.

2.1.2 Predicción Inter-fotograma

A diferencia del tipo de predicción anterior, éste no tiene exclusivamente en cuenta el fotograma actual. En vez de buscar la redundancia espacial, se intentará buscar una redundancia temporal.

Para aprovechar esta redundancia temporal en estándares anteriores los frames se dividían en tres clases distintas: I-frame, B-frame y P-frame que componen lo que se conoce como Group of Pictures (GOP).

La decodificación de un I-frame se lleva a cabo sin tener que recurrir a información proporcionada por ningún otro frame, mientras que tanto P-frames como B-frames necesitarán utilizar información proporcionada por otros frames en su decodificación. En el caso de los P-frames se obtendrá información únicamente de frames anteriores en el vídeo, a diferencia de los B-frames que harán uso de información de frames anteriores y posteriores. Por tanto la tasa de compresión de los B-frames es superior a la de los P-frames, siendo la de los I-frames la menor de las tres.

Como hemos comentado antes H.264 presenta un nuevo término denominado slice. De tal manera, que ahora en vez de hablar de I-frame, B-frame y P-frame, se hablará de I-slice, B-slice y P-slice. Al mismo tiempo se incorpora la posibilidad de utilizar varios I-slices (hasta un máximo de 16) ya codificados a la hora de realizar la predicción. Lógicamente esto puede ayudar a conseguir una mejor tasa de compresión con la contrapartida de aumentar el tiempo de codificación.

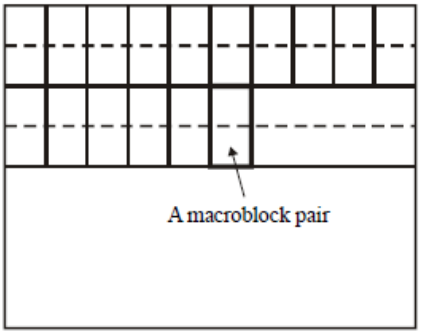

Figura 9. Macrobloque dentro de un frame **[9]**.

Como se puede apreciar en la Figura 9 los fotogramas se dividen en *slices* y cada slice a su vez contiene un número entero de pares de macrobloques. Existen varios modos de predicción según el número de bloques.

Las posibilidades a la hora de dividir un *macrobloque* en subbloques y las distintas dimensiones que éstos pueden adoptar dan un nuevo enfoque conocido como *tree structured*.

El *macrobloque* puede ser de 16×16 o de 8×8. Dentro del primero existen las siguientes dimensiones para subbloques: Inter16×16 (*macrobloque* igual a subbloque), Inter16×8 e Inter8×16 (en ambos casos el *macrobloque* se divide en dos subbloques de estas dimensiones) y por último Inter8×8 (el bloque se particiona en cuatro subbloques de 8×8). Para el segundo, se tienen también cuatro posibilidades: subbloque 8×8, dos subbloques 8×4 cada uno, dos subbloques 4×8 cada uno, o 4 subbloques de 4×4. [10]

Los *macrobloques* o subbloques se predicen a partir de un *macrobloque* o subbloque (de iguales dimensiones e igual o similar contenido) de un fotograma de referencia. La diferencia entre las coordenadas de ambos en sus respectivos fotogramas es lo que se conoce como el vector de movimiento (*motion vector según la nomenclatura común)*. Esta predicción es lo que se conoce como *motion estimation*. [11]

Codificar un *motion vector* para cada bloque puede tener un coste significativo en cuanto a tamaño, este coste aumentará cuanto menores sean las dimensiones de los bloques escogidos.

Los m*otion vectors* de bloques vecinos guardan relación entre sí, por lo que se pueden predecir unos a partir de otros ya codificados.

La diferencia entre el vector que se predice y el real se conoce como *Motion Vector Difference* o MVD y también debe ser codificada. De igual manera, habrá que tener en cuenta la diferencia entre el bloque estimado a partir del fotograma de referencia y el real. Para ello, se realiza la diferencia entre el bloque real y el estimado. Esta sustracción da lugar a un *macrobloque* residual que como ocurría con la diferencia anterior deberá ser también codificado.

En anteriores estándares se había introducido el concepto de subpíxel (en concreto half pixel), en este estándar se afina aún más, llegando al quarter pixel (qpel). Ahora un ejemplo de motion vector podría ser (0.75, -0.5). Obviamente las muestras necesarias para la predicción no existen en estas posiciones por lo que hay que calcularlas mediante interpolación. Lógicamente hacen falta más bits a la hora de representar tanto la parte decimal como la parte entera en el motion vector que si consideramos representar sólo la parte entera. En contrapartida utilizar motion vectors menos precisos (sólo con parte entera) obliga a tener más bits para los bloques residuales que si se utilizan motion vectors más precisos (parte entera y parte decimal).

Para llevar a cabo la interpolación hasta este nivel, se realiza primero una interpolación a nivel de half pixel en lo que luminancia se refiere. Para ello se utilizará horizontal y verticalmente un filtro FIR (Respuesta finita al impulso).

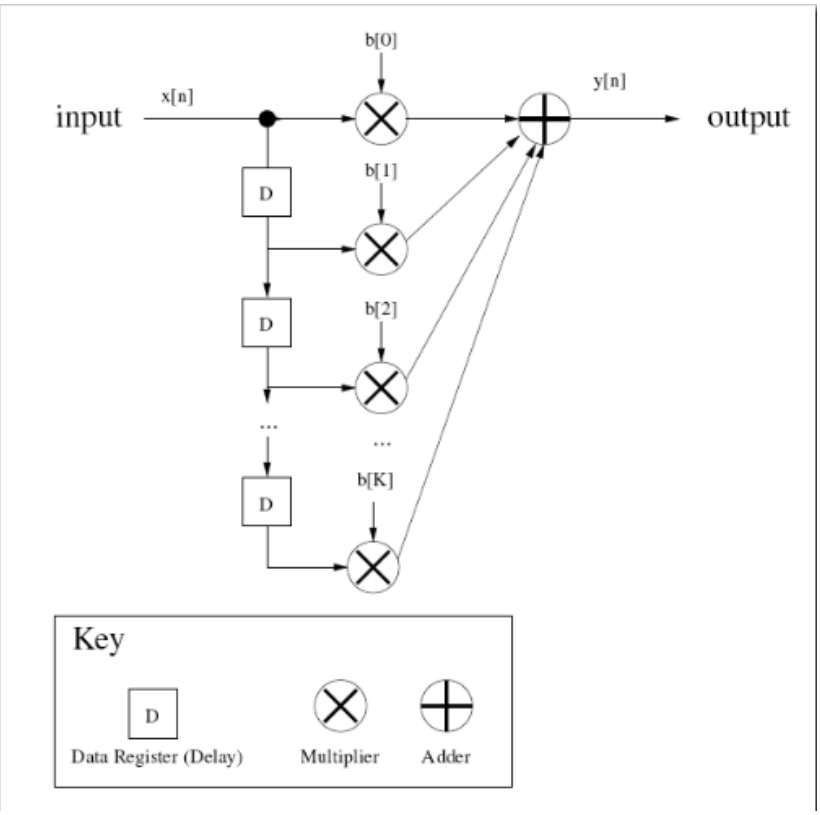

Figura 10. Taps dentro de filtro FIR **[12]**.

El nombre FIR viene de la manera en que el filtro afecta una señal. Una función impulso es una entrada, donde la señal es 0 siempre, excepto en un punto en el que es 1. La respuesta de un filtro FIR a esta entrada es de duración finita, es decir alcanza el valor 0 en un tiempo finito. El número de coeficientes que conforma la señal de salida se conoce como taps, de tal manera que un filtro con k + 1 coeficientes tiene un orden k. Los coeficientes multiplican las entradas después de un retardo y finalmente todas las salidas se suman componiendo la función de salida. [13].

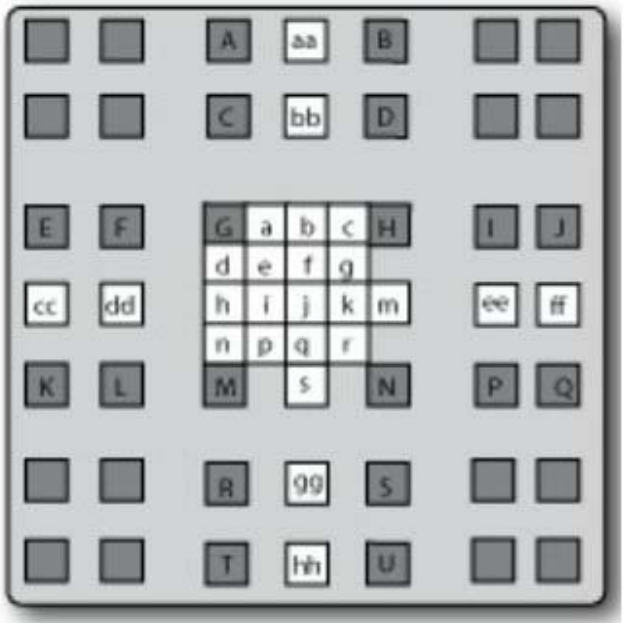

Figura 11. División y cálculo de diferentes niveles de subpíxel **[14]**.

Dicho filtro se puede expresar mediante la siguiente fórmula (cuyo resultado se redondeará a un entero), en función de los valores que se observan en la imagen anterior:

$$b = (E - 5F + 20G + 20H - 5I + J)/32\text{d} \qquad (1)$$

Es decir que el half pixel b se calcula a partir de las muestras de pixel que aparecen en la fórmula anterior (en este caso la interpolación es horizontal). En el resto de casos la interpolación se realizará a partir de otros half-pixel, una vez se ha completado la interpolación de los primeros. En la imagen anterior aparecen los half pixel: aa, bb, b, cc, dd, h, j, m, ee, ff, s, gg, y hh.

Cuando ya se ha completado la interpolación a nivel de half pixel (half-pel), se realiza la interpolación a nivel de qpel. Esta interpolación (en este caso se usa un filtro de media) se realiza primero sobre los qpel que tienen adyacentes horizontal o verticalmente, bien muestras half-pel (ya interpolados) o pixel. Después se llevarán a cabo las interpolaciones del resto de los qpel

(también usando un filtro de media) a partir de los half-pel que forman parte de la misma diagonal que ellos. Los qpel de la imagen anterior son: a, c, d, e, f, g, i, k, n, p y q. [9]

En lo visto hasta ahora, todas las operaciones se han realizado sobre los píxeles de la imagen, es decir se han llevado a cabo en el "dominio espacial" [15]. Se puede aplicar una transformada a los fotogramas de manera que pasemos del dominio espacial al dominio de la frecuencia.

En este estándar se utiliza una variación de la transformada del coseno (DCT). Si aplicamos esta transformada a un bloque de 4 × 4, en vez de 16 píxeles como tendríamos en el dominio espacial, tendremos 16 DCT coeficientes. Sin embargo, para muchos bloques un número importante de coeficientes poseen un valor de magnitud pequeño, por lo que se pueden cuantizar de manera conjunta o pueden ser descartados (posible pérdida). Además de esta transformada para bloques 4 × 4 y 8 × 8, se aplica la transformada de Hadamard para bloques 4 × 4 y 2 × 2.

El proceso de cuantización en este estándar se realiza de manera escalar, es decir se pasa del dominio de los números reales al dominio de los números enteros. Para llevar a cabo este cometido se realiza un redondeo del valor decimal original, utilizando los parámetros Qstep y PF. Este último depende de la posición que ocupa el coeficiente en la matriz (bloque), mientras que el primero guarda relación con el ya mencionado QP, de tal manera que cada vez que éste aumenta en 6 unidades Qstep duplica su valor (2). El aumento en los dos últimos parámetros mencionados reduce el número de coeficientes finales y por tanto incrementa la posibilidad de pérdida. [16]

$$\mathrm{Y} = \mathrm{X}.\,\mathrm{round}(\mathrm{PF}/\mathrm{Qstep}) \qquad (2)$$

Un mensaje se compone de un número finito de símbolos pertenecientes a un alfabeto y cada uno de ellos tendrá una probabilidad asociada, que tiene que ver con las opciones de ser emitido. Cuanto más alta sea dicha probabilidad menos cantidad de información habrá contenida en el símbolo y por tanto menos bits se necesitarán para su codificación. La información media

contenida en una emisión completa es lo que se conoce como entropía, de tal manera que si los símbolos de dicha emisión son equiprobables la entropía es máxima, es decir se proporciona la mayor información media por símbolo. [15]

En lo concerniente a la codificación entrópica, H.264 utiliza dos clases distintas: codificación de longitud variable (CAVLC) y codificación aritmética (CABAC). La primera genera palabras de código de longitud desigual para los diferentes símbolos, siendo los más probables los que se codifican con un menor número de bits y los menos probables los que se codifican con un mayor número de bits. Mientras que en la segunda no existe una correspondencia uno a uno entre los símbolos y las palabras de código, sino que a un mensaje se le asigna una palabra de código aritmético. Esta palabra define un intervalo entre cero y uno que se hace más pequeño (aumentando el número de bits necesarios para representarlo) a medida que se van introduciendo nuevos símbolos en el mensaje. [17]

2.2 HEVC

High Efficiency Video Coding (HEVC o H.265), es el estándar de compresión más reciente desarrollado por el Joint Collaborative Team on Video Coding (JCT-VC). Dicho estándar está publicado como ITU-T H.265 por el ITU-T Study Group 16 – Video Coding Experts Group (VCEG) y como ISO/IEC 23008-2 por el ISO/IEC JTC 1/SC 29/WG 11 Motion Picture Experts Group (MPEG).

La primera versión del estándar HEVC finalizó en abril del 2013. En la segunda versión se incluían 3 extensiones: RExt, SHVC y MV-HEVC y vio la luz en octubre del 2014. La tercera y última versión incluía la versión para 3D finalizándose en febrero de 2015.

La finalidad con la que el estándar HEVC fue desarrollado era proporcionar dos veces la tasa de compresión de su predecesor (H.264/AVC). Si se ha alcanzado esta meta o no dependerá del contenido a tratar, así como de las opciones utilizadas para codificar,

Lo que sí se puede remarcar, en cuanto a lograr este objetivo se refiere, es que para calidades de vídeo idénticas, HEVC alcanza un *bitrate* cercano o igual a la mitad del proporcionado por H.264. Y para tamaños de vídeo con tamaños o *bitrates* iguales, HEVC proporciona una calidad de vídeo significativamente superior.

H.264 utilizaba hasta 9 direcciones distintas (en la predicción 4×4) para la predicción intra-fotograma, HEVC puede usar 35 diferentes, con ésto se añaden muchas más posibilidades con respecto al bloque de referencia a utilizar y se permite una mayor eficiencia en esta clase de predicción. A cambio, el coste en lo que al tiempo de codificación se refiere aumentará. [18]

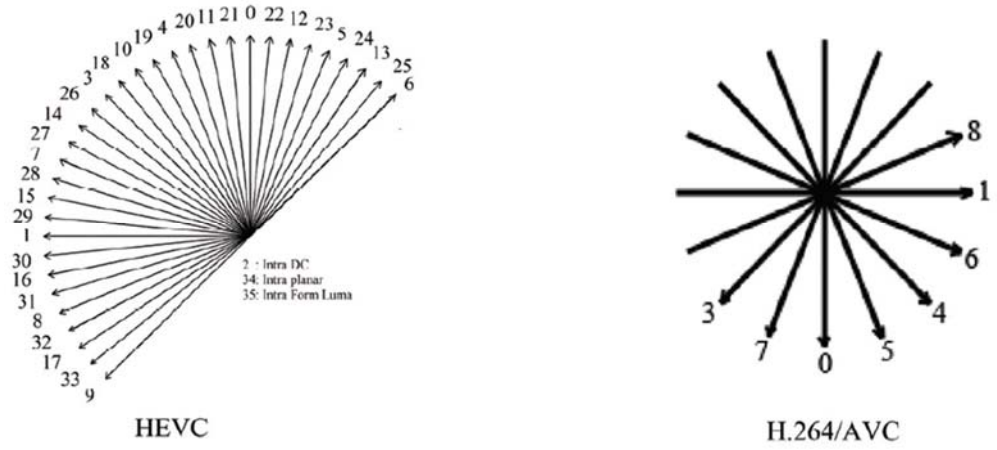

Figura 12. Direcciones en la predicción de HEVC y AVC **[19]**.

Aunque HEVC mantiene las divisiones en *slices*, mientras en H.264/AVC los macrobloques eran como máximo de 16×16 píxeles, HEVC introduce el término *Coding Tree Unit* (CTU) que sustituye a lo que en AVC se conocía como macrobloque y que puede alcanzar dimensiones más grandes llegando a un máximo de 64 × 64 píxeles. [20]

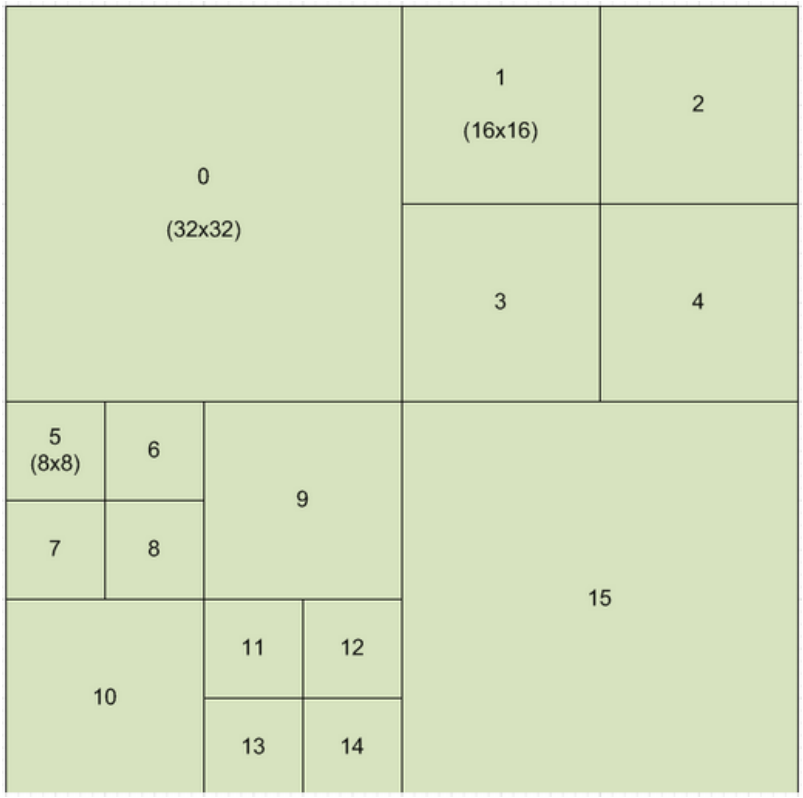

Figura 13. División en macrobloques y subdivisión en bloques en HEVC **[21]**.

Dependiendo de los parámetros utilizados en el códec, los macrobloques pueden tener diferentes dimensiones (64×64, 32×32 o 16×16). Como ocurría con el AVC, cada uno de ellos puede dividirse recursivamente, en este caso, la estructura de árbol es cuaternario y el tamaño mínimo de subbloque es 8×8. De tal manera que un CTB de 32×32 puede contener tres regiones de 16×16 y cuatro de 8×8. Estas regiones se denominan *coding units*, o CUs y son las unidades básicas de predicción del HEVC. 64×64, 32×32, 16×16 o 8×8 son las diferentes dimensiones que las CUs pueden adoptar. [22]

Con respecto al *motion vector* la granularidad se mantiene en qpel para este estándar, sin embargo se consigue un mejor comportamiento para HEVC al utilizar éste un filtro de 8 taps (por 6 del AVC) a nivel de *half pixel* y de 7 taps a nivel de qpel (en el AVC se utilizaba un filtro de media). [18]

En la etapa de la codificación entrópica HEVC sólo utiliza codificación aritmética (CABAC) y ya no utiliza codificación de longitud variable (CAVLC) como hacía su predecesor.

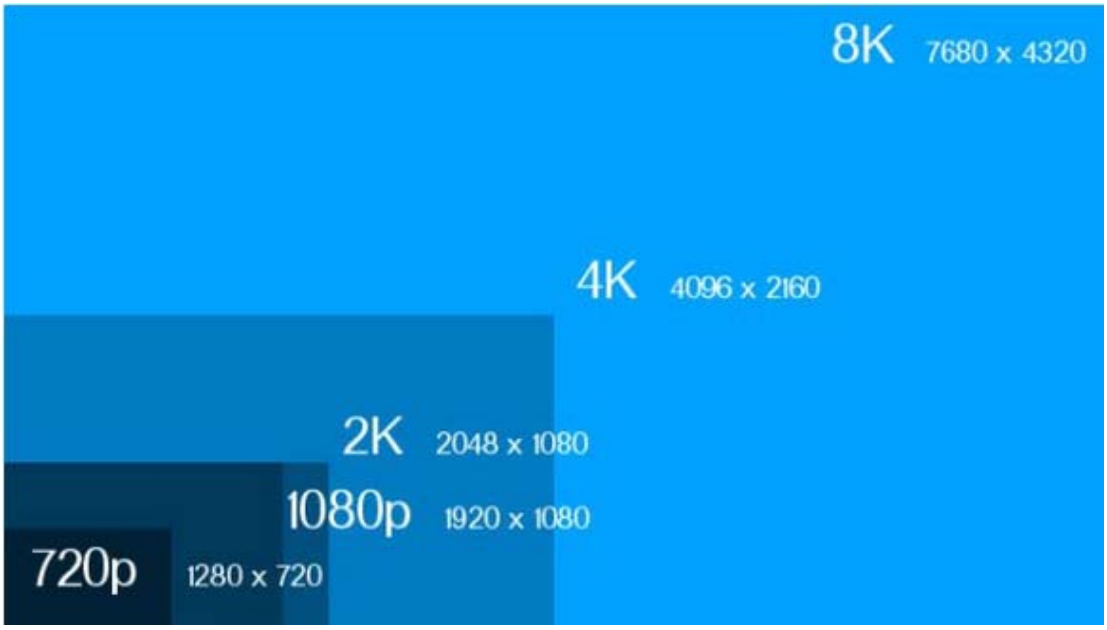

Figura 14. Distintas resoluciones de alta definición **[23]**.

En lo que tiene que ver con la resolución H.264 llegaba a soportar resoluciones de hasta 4,096 × 2,304 píxeles, es decir resoluciones superiores a 4K (4096 × 2160). El *framerate* máximo que AVC puede soportar son 59.94 fotogramas por segundo (fps). Para HEVC se alcanzan resoluciones superiores a 8K UHDTV (7680 × 4320) llegando a una resolución máxima de 8192×4320 píxeles. El *framerate* máximo que HEVC puede soportar son 300 fps. [24]

Existen dos implementaciones software fundamentales del estándar H.265: HEVC Test Model (HM) y x265. En ambas el vídeo que se recibe como entrada debe estar en formato *raw*. Suele decirse que un vídeo raw es un vídeo sin compresión, lo cual es cierto, éste no experimenta ningún tipo de compresión. Pero un vídeo sin compresión no es necesariamente *raw*.

Cuando una cámara está capturando imágenes la luz impacta sobre un sensor de imagen que convierte esta información en una combinación de ceros y unos. Hasta ese momento el vídeo

no ha sido interpretado, tarea que realiza el procesador de la cámara. Por tanto un vídeo raw debe ser interpretado mediante software si queremos visionar sus imágenes.

Un vídeo *raw* debe estar en uno de los siguientes formatos 4:2:0, 4:2:2 formato *planar*, 4:4:4 (Y'CbCr, RGB o GBR (igual a RGB pero con diferente intercalado)), o en formato raw 4:0:0. Si se utiliza una profundidad de bit superior a 8, se utilizarán 2 bytes que se almacenan en memoria de tal manera que el byte más significativo se almacena en una posición de memoria posterior a la del byte menos significativo (*little endian*). [25]

2.3 Modelos de color

2.3.1 RGB

En este modelo cada color aparece con tres componentes: rojo, verde y azul, al ser estos los colores primarios de la luz. La mezcla de estos colores primarios se conoce como síntesis aditiva. De la suma de colores luz se obtienen colores secundarios más luminosos que los colores primarios y de la ausencia total de estos tres colores el negro. En la representación digital del estándar RGB, cada uno de los componentes se codifican con 8 bits (sus valores van desde 0 hasta 255), teniendo un total de 24 bits. [26]

2.3.2 CMYK

El cyan, magenta y amarillo son los colores primarios de los pigmentos. El color pigmento refleja aquellas longitudes de onda que no ha absorbido de la luz del entorno.

Los pigmentos no tienen un color real objetivo, sino la capacidad molecular de absorber (sustraer) una parte de las ondas electromagnéticas de la luz y reflejar las restantes. Por ejemplo, al ser iluminado por luz blanca, el pigmento amarillo absorbe (sustrae) la luz azul y no la refleja. Cuando una sustancia refleja todo el espectro visible de la luz, su apariencia es blanca. A continuación se muestra la fórmula de conversión del modelo RGB al CMY (3) [27].

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad (3)$$

Sobre la fórmula anterior se puede comprobar lo descrito anteriormente, cuando tenemos un pigmento amarillo puro (Y = 1, C = 0, M = 0) no contiene azul, por lo que la B es igual a 0 mientras R y G están a 1.

El modelo de color CMYK es el más usado actualmente en la impresión a color. Las impresoras tienen un cartucho de color independiente de color negro. Es muy similar al modelo CMY, pero tienen un cuarto componente K que representa el color negro. Se puede pasar de un sistema a otro mediante las siguientes fórmulas (4) (5) (6) (7):

$$\mathrm{K} = \min(\mathrm{C}, \mathrm{M}, \mathrm{Y}) \qquad (4)$$

$$\mathrm{C} = \mathrm{C}\ \mathrm{K} \qquad (5)$$

$$\mathrm{M} = \mathrm{M}\ \mathrm{K} \qquad (6)$$

$$\mathrm{Y} = \mathrm{Y} - \mathrm{K} \qquad (7)$$

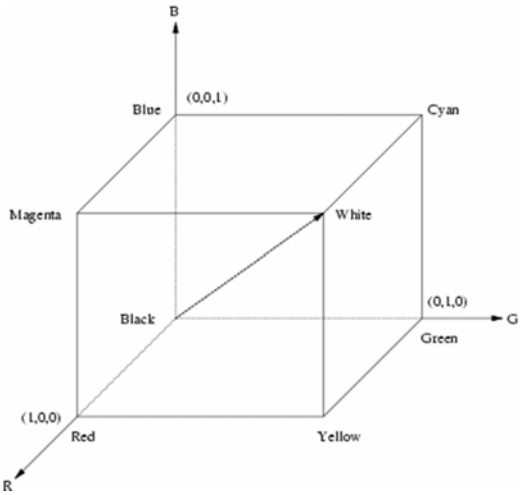

Figura 15. Cubo de relación entre los modelos de color RGB y YUV **[28]**.

2.3.3 YUV

La idea principal en la que se apoya este modelo de color consiste en que el sistema visual humano es más sensible al brillo (luminancia) para poder percibir los detalles y no tanto a la diferencia de color (crominancia). Se puede, por tanto, darle más importancia a la primera que a la segunda, plasmándolo en el número de bits utilizados en una y otra. Lo anterior no sólo supondrá un ahorro a nivel de ancho de banda, sino que se expresan las señales en consonancia con el funcionamiento del ojo humano.

El término YUV hace mención a una familia de espacios de color que siguen las premisas explicadas anteriormente, separar la información que tiene que ver con el brillo de la que tiene que ver con el color. YUV utiliza tres valores para representar cualquier color. Estos valores se caracterizan mediante las iniciales Y, U y V o mediante las iniciales Y', U, and V, dependiendo de si hablamos de luma (Y') o de luminancia (Y).

Aunque como se ha dicho YUV suele denominar a una familia de espacios que incluye a Y'CbCr, YCbCr (conocida popularmente como vídeo por componentes), YPbPr y otros, en realidad se refiere casi siempre al espacio de color Y'CbCr. El símbolo ' diferencia a la luma de la luminancia. Esta última se obtiene de los valores RGB lineales, mientras que la primera lo hace de los valores RGB no lineales ya que se le ha aplicado lo que se conoce como corrección gamma.

El ojo humano no tiene una respuesta lineal respecto a la luminosidad, a medida que aumenta la luminosidad, la percepción de tal luminosidad por parte del ojo humano no aumenta de manera proporcional. De hecho, como muchos sistemas biológicos, el HVS tiene una respuesta logarítmica, en este caso con respecto a la luminosidad. Un televisor o un monitor actual tienen también una respuesta logarítmica similar a la del ojo humano, se dice entonces que este dispositivo tiene una gamma asociada que habrá que corregir para mostrar fielmente la imagen original [29].

La luminancia es una medida más ajustada del brillo, mientras que la luma es más práctica en su uso por razones técnicas. A continuación se detalla la fórmula mediante la que se puede poner la luma en función del modelo de color RGB para un modelo de televisión estándar (especificación ITU-R BT.601).

$$Y' = 0.299R + 0.587G + 0.114B \quad (8)$$

Observando los coeficientes o pesos de la fórmula anterior, se puede comprobar cómo el ojo humano percibe de diferente manera el brillo para uno u otro color. El color azul aparece como el "menos brillante" y el color verde como el "más brillante. Para los nuevos modelos de televisión, que utilizan tecnología de alta definición (HD) se definió una nueva fórmula (9) en la especificación ITU-R BT.709 que se muestra a continuación.

$$\mathrm{Y}' = 0.2125R + 0.7154G + 0.0721B \quad (9)$$

Los componentes U y V, también se conocen como componentes de diferencia de color, ya que se obtienen restando el componente Y' del modelo de color YUV de los componentes B y R del espacio de color RGB. Por esto al componente U se le conoce también como proyección azul y al componente V como proyección roja (10) (11).

$$\mathrm{U} = \mathrm{B} - \mathrm{Y}' \quad (10)$$

$$\mathrm{V} = \mathrm{R} - \mathrm{Y}' \quad (11)$$

Dentro de este modelo de color existen diferentes alternativas en función de la proporción existente ente luminancia y crominancia, es lo que se conoce como subsampling. A continuación se definen las más importantes:

4:4:4 YUV: es el formato convencional, ya que no existe un descenso en lo que al muestreo se refiere. Los componentes Y, U (Cb), V (Cr) se muestrean para todos los píxeles. Cada componente necesita 8 bits, por lo que cada pixel requiere 24 bits. Este formato a menudo

se conoce como YUV sin especificar las cifras 4:4:4. En este caso se utilizan el mismo número de bits que si estuviéramos en el modelo RGB [30].

4:2:2 YUV: utiliza una proporción 2 a 1 horizontalmente. Por tanto el componente Y se muestrea para todos los píxeles, mientras que los componentes U (Cb) y V (Cr) se muestrean cada dos píxeles en dirección horizontal. Cada componente requiere 8 bits, por lo que cada par de píxeles necesitan 32 bits.

4:1:1 YUV: utiliza una proporción de 4 a 1 horizontalmente. Es decir que el componente Y se muestrea para todos los píxeles, mientras que lo componentes U (Cb) y V (Cr) se muestrean cada 4 píxeles horizontalmente. Por tanto, por cada 4 píxeles horizontales se requiere de 48 bits.

4:2:0 YUV: utiliza una proporción 2 a 1 horizontalmente y verticalmente. La componente Y se muestrea para todos los píxeles, las componentes U (Cb) y V (Cr) se muestrean para cada bloque de 2x2 píxeles. Como cada componente necesita 8 bits, cada bloque de 4 píxeles requiere 48 bits.

A la hora de almacenar la información 4:4:4 YUV ,4:2:2 YUV y 4:2:0 YUV lo hacen en formato *planar* o *triplanar*, es decir los componentes Y', U, y V se almacenan por separado en tres planos distintos (con dimensiones diferentes en función de las proporciones existentes entre luminancia y crominancia) [31], tal y como se aprecia en la Figura 16.

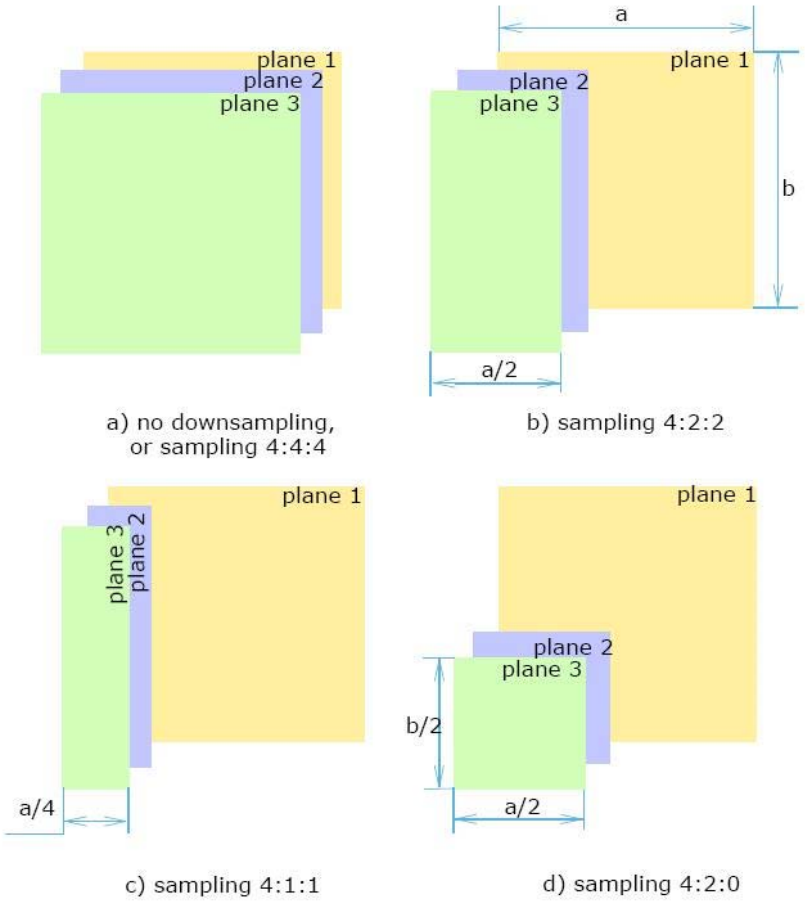

Figura 16. Distintos formatos de muestreo del modelo YUV **[32]**.

Si se utiliza este formato para el modelo de color RGB la información se almacenará tal y como aparece en la Figura 17.

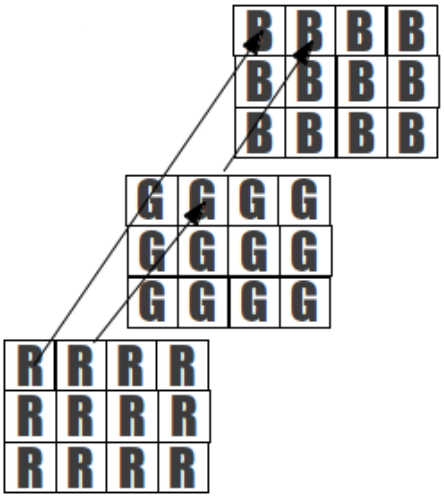

Figura 17. RGB, equivalente a YUV sin subsampling **[33]**.

Semiplanar o biplanar

Se utilizan dos planos en vez de 3, el primero contiene la información relativa a la luma y en el segundo aparecen de manera intercalada los dos componentes que tienen que ver con la crominancia, tal y como se muestra en Figura 18.

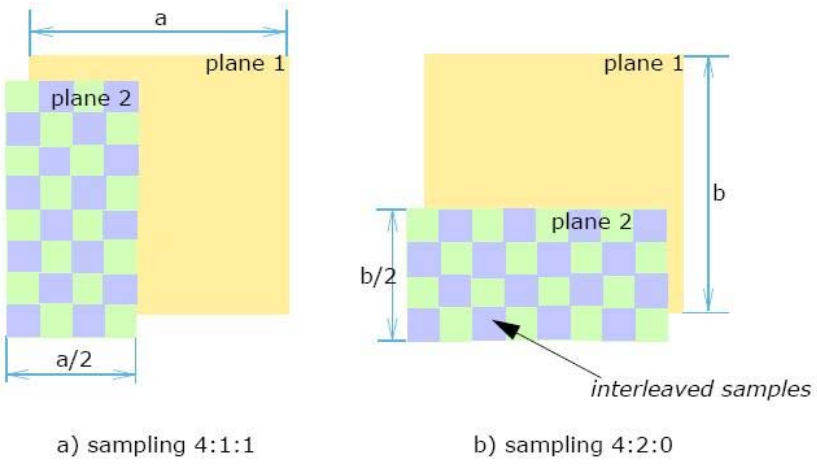

Figura 18. Subsampling con muestras intercaladas **[32]**.

Packed o pixel-oriented

En este formato los componentes de la información se almacenan en una array unidimensional en el que sus valores aparecen intercalados. Se puede ver la Figura 19 que muestra lo anterior para el modelo de color RGB.

Figura 19. Modelo RGB packed **[33]**.

2.4 VP9

A continuación, se presenta el estándar de compresión de vídeo VP9, principal competidor del HEVC. Este estándar es el sucesor del estándar VP8 que fue la alternativa al AVC y al igual que el VP8 es Open Source y fue desarrollado por la compañía Google finalizando su primera versión el 11 de junio de 2013. Tanto el navegador Google Chrome como el servicio de alojamiento de vídeos Youtube son capaces de soportar este códec.

WebM es el archivo contenedor de vídeo utilizado tanto para el códec VP8 como para el VP9. Sus dos características más relevantes son que es un formato implementado expresamente para la web y que no tiene ningún coste asociado.

La predicción intra fotograma en el VP9 es similar a la del AVC/HEVC, aunque en este caso hay 10 modos de predicción diferentes (de los que 8 son direccionales). En el VP9 este tipo de predicción requiere de dos arrays (vectores) unidimensionales que contienen los píxeles de los bloques vecinos que se sitúan encima y a la izquierda del bloque. El array que representa a los píxeles de la izquierda tiene la misma altura que el bloque actual, y el array que representa los píxeles superiores tiene de longitud el doble de la anchura del bloque actual. Sin embargo para bloques más grandes de 4×4, la segunda mitad del array guarda en todas las posiciones el mismo

valor que albergue la última posición de la primera mitad, como se puede observar en la siguiente imagen en la que el valor 80 se repite para toda la segunda mitad. [34]

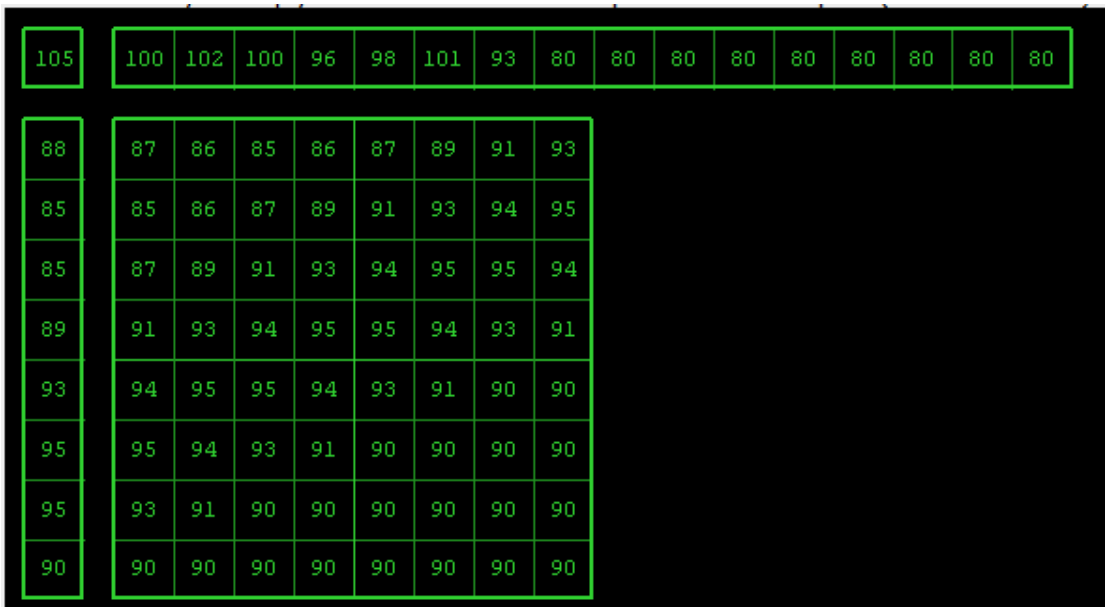

Figura 20. Ejemplo de predicción de VP9 para bloques superiores a 4×4 **[34]**.

En la predicción inter fotograma, algunas partes de la imagen pueden contener regiones de gran tamaño que se pueden predecir a la vez, por ejemplo un fondo de imagen, mientras que en otras partes se requiere un mayor nivel de detalle que además experimentan continuos cambios, para los primeros sería mejor opción utilizar bloques de gran tamaño, sin embargo para los segundos lo mejor sería utilizar bloques pequeños. VP9 proporciona la posibilidad de variar el tamaño de bloque, de tal manera que se tiene un rango de diferentes tamaños a aplicar según las características de la región.

El códec codifica cada imagen en superbloques de 64 × 64 píxeles. Cada superbloque tiene particiones que especifican como será codificado.

- Una partición de 64 × 64
- Dos particiones de 64 × 32
- Dos particiones 32 × 64

− Cuatro particiones de 32 × 32

Cada bloque 32 x 32 puede dividirse a su vez de manera similar a la anterior hasta llegar a una dimensión de 8×8. Las posibles combinaciones son las siguientes:

− Una bloque de 8 × 8
− Dos subbloques de 8 × 4
− Dos subbloques de 4 × 8
− Cuatro subbloques de 4 × 4

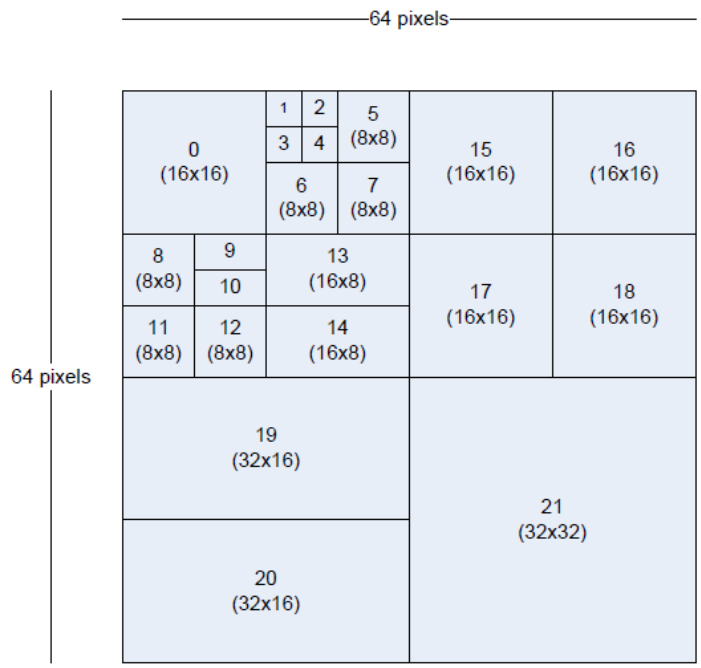

Figura 21. División en bloques y subbloques del VP9 **[35]**.

A la hora de llevar a cabo la transformadaVP9 es capaz de aplicarla de diversas formas, según las dimensiones usadas para la misma (4×4, 8×8, 16×16 y 32×32), así como del tipo de transformada a utilizar, que puede ser la DCT (transformada discreta del coseno) o la DST (Transformada discreta del seno). La elección de esto último viene ya determinada por el modo

elegido en la predicción intra fotograma. La elección de las dimensiones de la transformada a aplicar puede ser especificada a nivel de fotograma o a nivel de bloque. [35]

En lo que al motion vector se refiere si en HEVC llegábamos a una precisión de qpel, en el caso del VP9 se llega al 1/8th pel. Con respecto a los fotogramas de referencia a utilizar, en conjunto con el motion vector, existen tres tipos de fotogramas de referencia para este estándar: Last, Golden, o AltRef. Dicho motion vector se aplicará con granularidad 8 × 8, existiendo por lo general un solo motion vector por bloque, es decir que éste se aplica de manera unidireccional y no existe bipredicción. Sin embargo, VP9 soporta la llamada predicción compuesta en la que existen dos motion vector por bloque y se realiza un promedio sobre las dos predicciones resultantes. Con el objetivo de evitar patentes existentes sobre la bipredicción, la predicción compuesta sólo tendrá lugar en aquellos fotogramas que no van a ser visionados. [34]

Estos fotogramas son un problema cuando se almacena el bitstream en el contenedor WebM antes mencionado, ya que cada frame del contenedor debe ser un fotograma que pueda ser visionado. Por eso VP9 introduce el concepto de super-frame. Un super-frame se compone de uno o más fotogramas que no pueden ser visionados y uno que sí puede serlo, de tal manera que sean un único elemento dentro del contenedor.

En la Tabla 7 se muestra un resumen con las características más importantes de los códecs anteriormente descritos.

	H.264	**HEVC**	**VP9**
Transformada	DCT/ Hadamard	DCT/DST	DCT/DST
Intra predicción	8 modos	33 modos direccionales + planar + DC	10 modos
Tamaño de bloque	16×16	Hasta 64x64	32x32
Codificación entrópica	CABAC/VLC	CABAC	Codificación aritmética
Resolución	4.096×2.304	8.192×4.320	65.536×65.536
Espacios de color	YUV	YUB y RGB	YUV 4:2:0

Tabla 7. Resumen de códecs.

3. Metodología

Una vez se han explicado las etapas de distintos códecs, hay que añadir que se pueden incorporar a éstos dos etapas adicionales. Una se llevará a cabo antes de se inicie cualquier otra tarea (preprocesamiento) y otra se efectuará en último lugar (postprocesamiento).

El preprocesamiento suele consistir básicamente en la aplicación de diferentes filtros al vídeo original. Se puede, por tanto, llevar a cabo un filtrado que suavice las imágenes, elimine el ruido presente en las mismas o prescinda información desechable antes de que se lleve a cabo la codificación. Esto permitirá mejorar la calidad del vídeo inicial o facilitar la tarea de codificación. Se muestra en la Figura 22 el orden de las fases de la codificación vistas en el capítulo anterior, así como la etapa de preprocesamiento de la que se hablará a continuación.

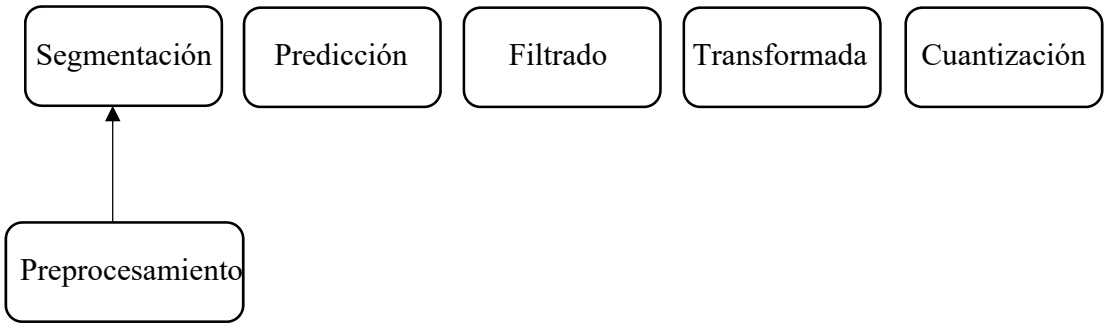

Figura 22. Fases de la codificación incluyendo preprocesamiento.

Para configurar la etapa de preprocesamiento habrá que elegir aquellos filtros (o combinaciones de éstos) que se utilizarán en la misma. Para llevar a cabo dicha elección se tendrá que tener en cuenta si las características del filtro contribuyen a la finalidad que se busca: recordemos que es disminuir el tamaño de bitstream sin perder calidad de forma apreciable.

3.1 Elección de filtro

Un filtro es un mecanismo que opera sobre una determinada señal de entrada para producir una señal de salida diferente [15]. En la sección anterior se hablaba de las técnicas utilizadas por los códecs y de cómo éstas a veces se empleaban en el dominio del espacio y otras en el domino de la frecuencia. Lo mismo sucede con los filtros, se pueden aplicar en ambos dominios, siendo la convolución en el dominio del espacio el equivalente a la multiplicación en el dominio de la frecuencia. Las operaciones realizadas en el dominio de la frecuencia se llevan a cabo esta vez en la transformada de Fourier de la imagen. [36], en este trabajo se aplicarán los filtros relativos al preprocesado en el domino del espacio.

3.1.1 Filtros lineales

Los filtros lineales están basados en las llamadas máscaras de convolución o *kernels*. Dentro del dominio espacial una máscara es una matriz de unas determinadas dimensiones que se aplica sobre una parte de la matriz que representa a la imagen (fotograma) a través del proceso de convolución.

La convolución consiste en multiplicar cada una de las posiciones de la región correspondiente del fotograma por la correspondiente de la máscara de convolución. Posteriormente se suman los valores obtenidos en cada una de las multiplicaciones efectuadas. El valor obtenido puede ser a su vez normalizado [37]. Se puede ver un ejemplo concreto de lo anterior en la Figura 23.

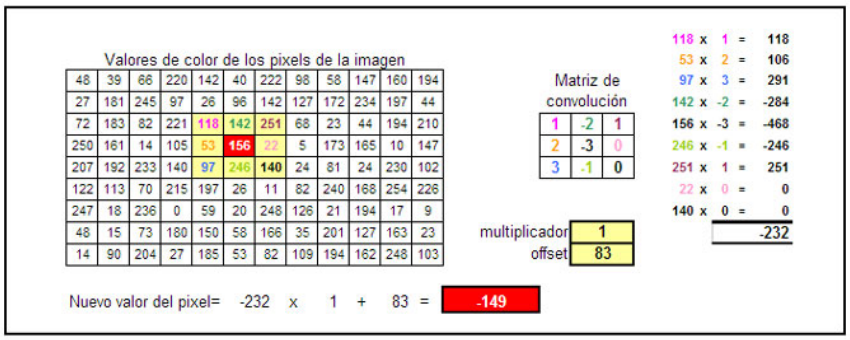

Figura 23. Ejemplo de cálculo de valor de pixel en un filtro lineal

3.1.1.1 Filtro de la media

El ruido es la información no deseada que contamina la imagen. El origen puede estar tanto en el proceso de adquisición de la imagen (errores en los sensores), como en el de transmisión (debido a interferencias en el canal de transmisión). De tal manera que la función que determina la imagen (representada por g(x,y)) se compone de la suma de la imagen original (representada por f(x,y)) y el mencionado ruido (representado por r(x,y)), tal y como aparece en la expresión 11. [36]

Dos de los principales tipos de ruido que pueden aparecer en una imagen son el ruido gaussiano y el ruido de sal y pimienta. Mientras el primero tiene su origen en el proceso de transmisión, el origen del segundo está en el proceso de adquisición de la imagen por parte del sensor de la cámara. Dicho sensor (que puede ser CCD o CMOS) está formado por celdas fotosensibles que incluyen uno o más fotodiodos que convierten la luz en electricidad. Esta actividad eléctrica, en sí misma, generará una cierta señal aún en ausencia de luz. Dicha señal oscilará en relación con la temperatura, generando datos al azar, que originarán ruido. [38]

El filtro de la media aritmética suaviza las variaciones locales dentro de una imagen, de esta manera se consigue tanto suavizar la imagen como eliminar el ruido (sobre todo cuando es ruido gaussiano) existente en la misma.

$$g(x,y) = f(x,y) + r(x,y) \quad (11)$$

En este filtro la máscara de convolución es una matriz cuyos elementos toman el valor uno. Las dimensiones habituales de la máscara de convolución son 3 × 3 ó 5 × 5.

3.1.1.2 Filtro de la mediana

Está dentro de los denominados filtros de orden, ya que a partir de la ordenación que se hace de unos valores obtenidos se obtiene el valor de salida. Para este filtro se toman

normalmente regiones con dimensiones 3 × 3 ó 5 × 5, con el fin de obtener un nuevo valor (valor de salida) para el pixel central de la misma. Dentro de la región habrá 9 ó 25 puntos, dependiendo de la dimensión de la misma. Si se ordenan estos valores (de menor a mayor), el valor que ocupe la posición central de forma cardinal será el valor de salida. El quinto en el caso de la dimensión 3 × 3 y el décimo tercero en el caso de la dimensión 5 × 5.

La principal función del filtro de mediana es hacer que los píxeles con intensidades distintas se hagan muy parecidos a sus píxeles vecinos, eliminando así los picos de intensidad que aparezcan aislados en el área de la máscara. Este filtro elimina de manera eficiente el ruido denominado de sal y pimienta.

El filtro de mediana podría ser una buena elección con objeto de realizar la labor de preprocesado. Como se expuso anteriormente, este filtro es capaz de suavizar la imagen y eliminar su ruido (especialmente si se trata de ruido de sal y pimienta). Además este filtro a diferencia del filtro de media preserva los bordes de las figuras contenidas en la imagen.

No sólo se podría experimentar una mejora en cuanto a calidad si el vídeo original contuviera ruido, sino que se favorece la codificación. Como se exponía anteriormente una de las predicciones realizadas en la codificación tiene que ver con los valores dentro de un mismo fotograma (intra-predicción), cuanto más se parezcan las distintas zonas que lo componen, más eficientemente se realizará dicha predicción (localidad espacial)..

Por otra parte, es probable que dos fotogramas distintos de un vídeo se parezcan más una vez se han eliminado mediante filtrado aquellos picos de intensidad que más pueden diferenciarles, favoreciendo en este caso la inter-predicción (localidad temporal).

Para la fase de la cuantización será deseable también tener el menor número de valores diferentes en dominio del espacio, ya que esto derivará en un menor número de coeficientes en el dominio de la frecuencia. El filtro de la mediana también contribuye a este objetivo.

3.2 Filtro-Operador

Para comprobar si este preprocesamiento mejora el resultado obtenido por el códec, se puede medir la calidad de dicha salida al aplicar el preprocesamiento y compararla con la salida que se obtiene sin aplicarlo mediante varias métricas que se comentarán posteriormente.

Los bordes de una imagen digital se pueden definir como transiciones entre dos regiones con valores significativamente distintos. Suministran una valiosa información sobre las fronteras de los objetos y ésta puede ser utilizada para segmentar la imagen o reconocer objetos dentro de la misma [39]. Una vez realizada la elección anteriormente expuesta, se puede ver si es posible mejorarla haciendo que se preserven aún mejor los bordes de las figuras contenidas en la imagen.

Cuando hablamos de funciones y queremos medir su tasa de variación, utilizamos el operador diferencial o derivada en el caso de ser una función unidimensional. Si interpretamos una imagen como una función bidimensional (ya que depende de dos coordenadas) f(x,y) la derivada es un vector que apunta en la dirección de la máxima variación de f(x,y) y cuyo módulo es proporcional a dicha variación. Este vector se denomina gradiente, se puede ver como se calcula dicho gradiente y su magnitud en las siguientes ecuaciones (12) (13):

$$\nabla f(x,y) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix} \quad (12) \qquad Mag\big(\nabla f(x,y)\big) = \sqrt{\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2} \quad (13)$$

Al hablar de imágenes, nuestra variable (la luminosidad) es discreta, sus valores pertenecen al conjunto que va desde cero hasta 255 (si se tiene una profundidad de pixel de 8 bits). Para una variable discreta es posible aproximar las fórmulas de gradiente a través de operadores numéricos de primer orden (14) (15).

$$\frac{\partial f(x,y)}{\partial x} \approx \nabla_x f(x,y) = f(x,y) - f(x-1,y) \qquad (14)$$

$$\frac{\partial f(x,y)}{\partial y} \approx \nabla_y f(x,y) = f(x,y) - f(x,y-1) \qquad (15)$$

Por tanto las versiones más sencillas de las máscaras de convolución que se pueden aplicar serían $[-1 \quad 1]$ para la primera aproximación y $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$ para la segunda. Estas máscaras destacan por su sencillez, sin embargo con estas aproximaciones se calcula el gradiente entre dos posiciones. De hecho el cálculo del gradiente se realiza sobre la posición intermedia que existe entre las dos. Por este motivo se añadió un 0 entre los dos valores existentes en ambas máscaras.

Al utilizar filtros de una sola fila o una sola columna como máscara se apreciaban altos niveles de ruido en los resultados obtenidos, por ello se decidió aplicar filtros con tres filas y tres columnas que disminuían el nivel de ruido. Una vez aplicadas las máscaras de convolución se obtienen los gradientes con respecto a *x* y con respecto a *y*, para después obtener magnitud del gradiente según la métrica euclídea usual.

Los operadores que se describen a continuación tienen distintas máscaras de convolución, pero todos están basados en los principios descritos anteriormente.

3.2.1 Operador de Roberts

Obtiene buena respuesta ante bordes diagonales. Ofrece buenas prestaciones en cuanto a localización. El gran inconveniente de este operador es su extremada sensibilidad al ruido y por tanto tiene pobres cualidades de detección. Sus máscaras de convolución (16) (17) se muestran a continuación [39].

$$G_x = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \qquad (16)$$

$$G_y = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (17)$$

3.2.2 Operadores de Sobel y Prewitt

Estos dos operadores tienen la propiedad añadida de suavizar la imagen, eliminando parte del ruido y minimizando la aparición de falsos bordes debido a la magnificación de este [15]. El primero es un filtro diseñado para que sea más sensible que el de Prewitt a los bordes diagonales. Las máscaras de convolución de dicho operador son las siguientes (18) (19):

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad (18)$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad (19)$$

El operador de Prewitt tiene un comportamiento similar al de Sobel, aunque en este filtro se involucran a los vecinos de filas y columnas adyacentes para proporcionar una mayor inmunidad al ruido. Sus máscaras de convolución son las siguientes (20) (21):

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad (20)$$

$$G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad (21)$$

Tal y como se ha comentado anteriormente, las máscaras de convolución debén ser normalizadas. La normalización en el filtro de Sobel es 1/8, mientras que en el filtro de Prewitt es 1/6. Se pueden formular de manera conjunta ambas máscaras (22) (23) de la siguiente forma:

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ k & 0 & -k \\ 1 & 0 & -1 \end{bmatrix} \quad (22) \qquad G_y = \begin{bmatrix} -1 & -k & -1 \\ 0 & 1 & 0 \\ 1 & k & 1 \end{bmatrix} \quad (23)$$

Los valores de *k* varían de un filtro a otro, siendo *k* igual a uno para el operador Prewitt y dos para el operador Sobel.

3.2.3 Aproximación del threshold

Se ha desechado el operador de Roberts al tener peor comportamiento que los otros dos (de similares características), eligiendo finalmente el operador de Sobel. Una vez elegidos operador y filtro, la manera de proceder sería no filtrar aquellas zonas en las el operador aprecie bordes y sí hacerlo en zonas cuyos píxeles no formen parte de bordes.

Cuando se calcule la magnitud del gradiente habrá que fijar un valor a partir del cual consideremos que el pixel forma parte de un borde (threshold). Una forma de hacerlo es realizar el cálculo empíricamente para cada secuencia de vídeo que se tenga como entrada.

Como estrategia, se puede asignar un determinado threshold y aplicar el operador correspondiente, para comprobar después el número de píxeles identificados como partes de un borde. Cuanto mayor sea dicho threshold menos píxeles se identificarán como integrantes de un borde, de igual manera cuanto menor sea su valor más píxeles se identificarán como partes de un borde. Teniendo lo anterior en cuenta se puede ir modificando el threshold hasta ajustarlo de forma que los píxeles identificados por el operador coincidan de la manera más exacta posible con los que realmente forman parte de un borde.

Mediante ciertas herramientas de software como Video Quality Caliper [6] se pueden conocer los valores de luminancia y crominancia de los píxeles de un fotograma a la vez que se

visualiza dicho fotograma. Esto permite conocer el valor del gradiente asociado a cada uno de los píxeles después de aplicar la detección de bordes.

Detectar el menor valor de gradiente de un pixel que integra un borde proporcionará el threshold más adecuado. Los píxeles que tengan un gradiente asociado menor a ese threshold no pertenecerán a bordes y por tanto no hará falta que sean preservados. Se presenta en la Figura 24 un diagrama que ilustra el proceso de aproximación al threshold ideal que se ha visto hasta el momento.

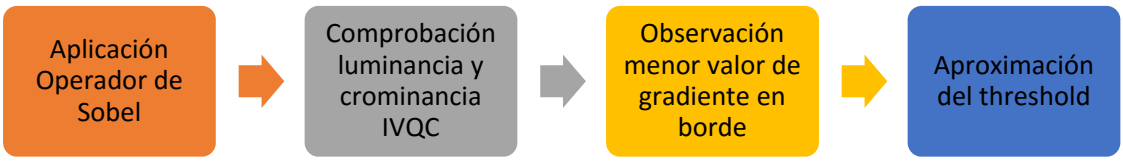

Figura 24. Primer proceso de aproximación al valor de threshold ideal.

Para acercarse a ese valor se puede ver también el porcentaje de píxeles de la imagen que superiores o inferiores al threshold, en principio en todos los fotogramas de un vídeo deben predominar el número de píxeles que no forman parte de los bordes sobre el número de aquellos que sí integran dichos bordes.

Se ha implementado en un lenguaje de alto nivel (Ansi C) el operador de Sobel en combinación con el filtro de mediana (filtro combinado). Teniendo como parámetros: la resolución (ancho y alto de los fotogramas), el subsampling y el threshold, sólo se filtrarán aquellos puntos cuyo gradiente (calculado por el operador) esté por debajo del valor del threshold.

Posteriormente se ha modificado el código fuente de dicha implementación para que se contabilicen el número de píxeles que están por encima de un valor (threshold) y los que están por debajo en cada fotograma. De esta manera será posible realizar una media aritmética de los resultados obtenidos para calcular el porcentaje de píxeles filtrados.

Es posible que uno o varios de los fotogramas que forman parte del vídeo presenten unas determinadas características que puedan ser de ayuda a la hora de llevar a cabo la aproximación del threshold. Por ejemplo para un fotograma en el que no aparece ninguna figura el número de píxeles detectados como integrantes de bordes debería tender a 0 si el threshold elegido es el adecuado. Todo este segundo proceso se representa en el diagrama de la Figura 25.

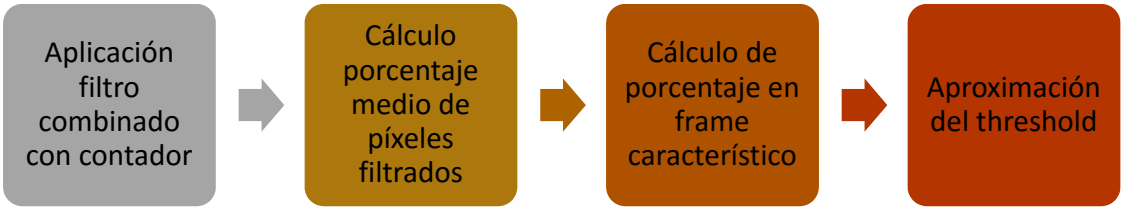

Figura 25. Segundo proceso de aproximación al valor de threshold ideal.

A modo de ejemplos ilustrativos de la metodología mostrada se han elegido dos vídeos en formato raw con una resolución de 176 × 144 y subsampling 4:2:2 y 4:2:0 respectivamente (obtenidos desde la web https://media.xiph.org/video/derf/).

3.2.3.1 Video 1

El primero de los vídeos (football_422_qcif.yuv) está compuesto por 360 frames y tiene 4:2:2 como subsampling.

- **Primer Proceso**

A continuación se muestran dos imágenes en las que se muestra el mismo fotograma del vídeo elegido, antes y después de aplicarle el operador de Sobel. En ambos fotogramas se aprecian los valores de luminancia (102 en el primero y 5 en el segundo) y crominancia del pixel con coordenadas $x = 92$ e $y = 110$. Dicho pixel forma parte del fotograma que se puede considerar como el fondo del mismo (la hierba del terreno de juego) y por tanto no forma parte de ningún borde, sin embargo el valor del gradiente asociado al mismo es mayor que el threshold y se estaría identificando como integrante de un borde, a la hora de combinarlo con el filtro de mediana este filtro no se le aplicaría. Es necesario por tanto aumentar el valor del threshold.

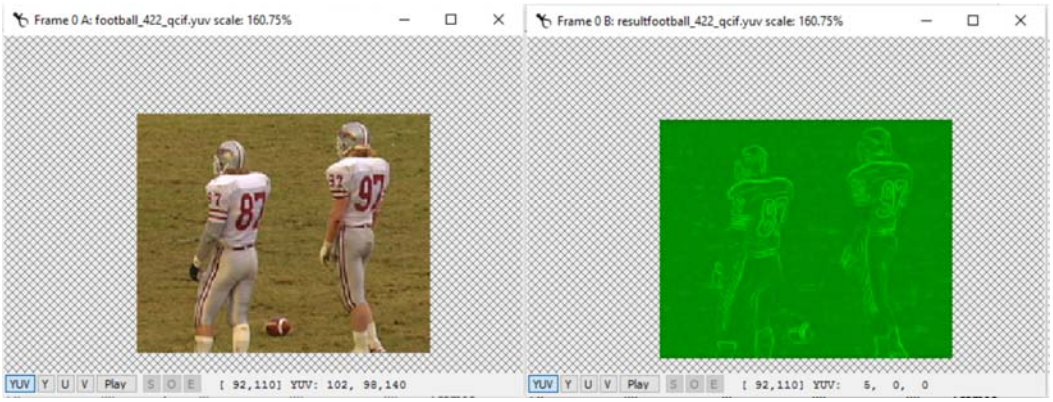

Figura 26. Frame del primer vídeo antes y después de aplicarle el operador de Sobel.

- **Segundo Proceso**

La multiplicación del ancho (176) por el alto (144) da como resultado 25344 por lo que al ser el subsampling 4:2:2 se tienen 50688 valores entre crominancia y luminancia. Después de efectuar la detección de bordes mediante el operador de Sobel implementado con un threshold igual a 0 (el mínimo posible) se ha realizado una media aritmética de los píxeles detectados para la totalidad de los fotogramas (360). Obviamente el resultado se aleja mucho de la idea teórica antes indicada (predominio de número de píxeles que no integran bordes sobre número de píxeles que sí integran bordes) al detectarse como bordes el 73 % de los píxeles.

Con un threshold igual a 0 el fotograma número 294 marca el máximo número de píxeles no integrantes de bordes con 32844, este número no refleja la información de cambios de contorno de forma acertada, ya que dicho fotograma no contiene ninguna figura. Se muestra en la Figura 27 el fotograma después de aplicarle el operador de Sobel.

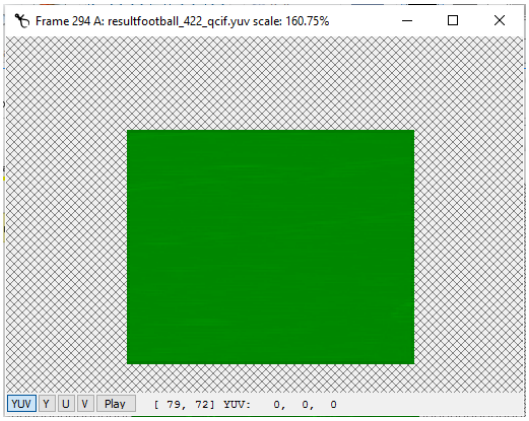

Figura 27. Fotograma después de aplicarle el operador de Sobel con threshold igual a 0.

- **Combinación de procesos (Proceso híbrido)**

Si el valor elegido como threshold es 15, la tasa de píxeles detectados (como no integrantes de bordes) asciende a un 85 % (se detectan haciendo el promedio entre todos los fotogramas obteniéndose 43484 píxeles por fotograma), dato mucho más cercano a la realidad. Visualizando algunos fotogramas se ha comprobado que hay píxeles que forman parte de determinados bordes cuyos valores se acercan a ese threshold, por lo que se puede concluir que éste es un threshold aceptable a la hora de combinar el filtro de mediana con el operador de Sobel para este vídeo.

3.2.3.2 Vídeo 2

En este segundo ejemplo se tiene una entrada diferente (mother_daughter_qcif.yuv descargado desde https://media.xiph.org/video/derf) compuesta por 300 frames y con misma resolución que la anterior, pero con 4:2:0 como subsampling (pasamos por tanto de 50688 valores a 38016).

- **Segundo Proceso**

La tasa de píxeles que no forma parte de bordes no es tan baja en este caso como lo era en el ejemplo anterior. Si se aplica un threshold con valor igual a 0 se obtiene una tasa 36 %, valor de nuevo que difiere de la realidad.

Como ocurría anteriormente, puntos del fondo de la imagen (en este caso la pared) se consideran integrantes de bordes de figuras de la imagen. Se muestra a continuación uno de los fotogramas originales del vídeo y el resultado obtenido al aplicarle el operador de Sobel:

Figura 28. Frame del segundo vídeo antes de aplicarle el operador de Sobel.

- **Combinación de procesos (Proceso híbrido)**

Se identifican algunos píxeles con valores muy bajos (inferiores a 10) en los bordes que coinciden con el cabello de las figuras. Seleccionando como threshold un valor igual a10 se obtiene como resultado 32331 píxeles de los 38016 totales como no integrantes de ningún borde de figura. El porcentaje ha cambiado desde un 36% (con threshold igual a 0) a un 85% (con un threshold igual a 10). En este ejemplo, es el primer fotograma el que refiere el máximo de píxeles fuera de los bordes. En este aspecto existe mucha más homogeneidad, ya que no existe apenas movimiento en el vídeo y todos los fotogramas son semejantes entre sí.

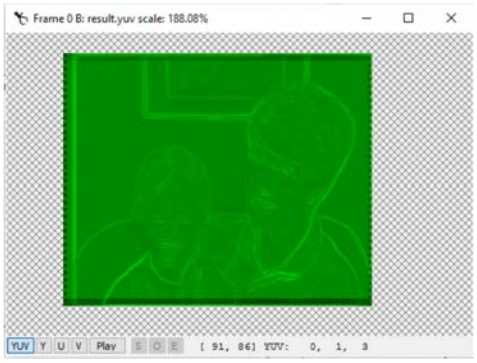

Figura 29. Frame del segundo vídeo después de aplicarle el operador de Sobel.

3.3 Métricas de calidad

Para validar los experimentos que se realizarán, es necesario llevar a cabo mediciones de los tamaños de los bitstreams y de la calidad de los vídeos reconstruidos. Las métricas de calidad de vídeo (VQM en inglés) realizan una comparación entre el vídeo original, sin codificar, y el vídeo reconstruido mediante decodificación a partir del bitstream de la codificación. Estas métricas se dividen en subjetivas (basadas en resultados ofrecidos por observadores humanos) y objetivas (basadas en métodos matemáticos).

3.3.1 Medidas objetivas

3.3.1.1 PSNR

En esta medida objetiva el vídeo reconstruido viene representado por la función F(x,y), mientras que el vídeo original lo representa la función f(x,y). Mediante dichas funciones se calcula el error cuadrático medio (MSE), básicamente consiste en restar a la luma de un pixel en una determinada coordenada del vídeo original el valor de esa luma en esa misma posición del vídeo reconstruido (cuando se está en el espacio de color YUV). Una vez hecho esto, se realiza el sumatorio de todas estas restas elevadas al cuadrado y se divide el resultado obtenido entre la resolución del vídeo original.

Es decir, para un vídeo con resolución N×N quedarían las siguientes fórmulas (24) (25) donde RMSE indica la realización de la raíz cuadrada sobre la MSE:

$$MSE = \frac{\sum[f(i,j)-F(i,j)]^2}{N^2} \qquad (24) \qquad PSNR = 20\, log_{10}\left(\frac{255}{RMSE}\right) \qquad (25)$$

Se utiliza la escala logarítmica y el valor 255 del numerador de la segunda fórmula viene dado por la profundidad de bit de los fotogramas del vídeo, por lo general dicha profundidad suele ser 8. La unidad en la que se mide la PSNR es el decibelio.

3.3.1.2 SSIM (Índice de similitud estructural)

En este caso la medición de calidad tiene como factor fundamental la similitud estructural entre el vídeo de referencia (original) y el reconstruido.

Se conoce como imágenes naturales a aquellas tomadas mediante una cámara en un determinado entorno. Las imágenes naturales están altamente estructuradas, es decir que sus píxeles experimentan una gran dependencia ente sí, especialmente cuando están próximos [40]. Las dependencias existentes conllevan información esencial sobre las estructuras de los objetos en la escena.

La información estructural es independiente de la iluminación, de forma que la influencia de ésta debe de ser aislada en el cálculo de la calidad de la imagen. Por lo tanto se calculan tres componentes de forma independiente y por separado: luminancia, contraste (e) y estructura (s) [41]. Las variaciones de luminancia y contraste no producen una distorsión en la información estructural de las imágenes y aunque deben formar parte de la métrica final, habrá que separar dentro de ésta las variaciones que sufren las imágenes en variación de luminancia, de contraste y estructurales [42]. Dicha separación se puede apreciar en la Figura 30.

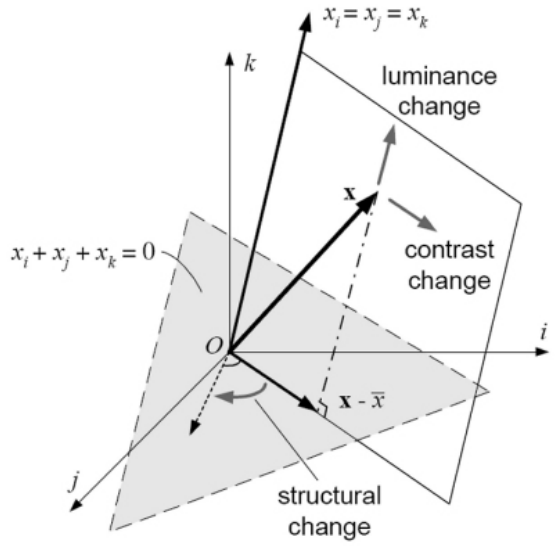

Figura 30. Factores de variación que pueden producir distorsión **[42]**.

El origen representa la imagen original y los vectores de variación de luminancia y contraste sobre dicha imagen original definen un plano. La variación estructural viene dada por la rotación de este plano sobre su eje en un ángulo que determinará el valor de la variación.

La fórmulas (26) (27) (28) mediante las que se calculan las variaciones de cada uno de los factores que inciden en el valor de esta métrica son las siguientes:

$$l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \quad (26) \qquad c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} \quad (27) \qquad s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3} \quad (28)$$

Donde μ_x representa la media de x, μ_y la media de y, σ_y^2 la varianza de y, σ_x^2 la varianza de x y σ_{xy} la covarianza de x e y. Las constantes c_1, c_2 y c_3 se utilizan para no tener un denominador cercano a cero, siendo $c_3 = \frac{c_2}{2}$ y siendo c_1 y c_2 el producto entre el valor máximo alcanzable en función de la profundidad del pixel (normalmente 255) y una constante cuyo valor tiende a cero.

La expresión de la métrica SSIM (29) en función de los valores de las fórmulas anteriores es la siguiente:

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma} \quad (29)$$

En la fórmula anterior α, β y γ son siempre mayores que cero y sirven para ajustar el peso relativo de cada uno de los tres componentes. Su valor final es un número decimal ente -1 y 1 [42].

Se entiende que las imágenes naturales son aquellas a las que, por su distribución estadística, el HVS (Human Visual System) está mejor adaptado. Por ello esta métrica se asemeja más a como un observador humano percibe la calidad de la imagen que la considerada previamente (PSNR).

4. Análisis Experimental

Mediante la herramienta Video Quality Caliper se puede evaluar la calidad de la compresión de contenidos multimedia en formato de vídeo, en concreto en el estudio con dicha herramienta se utilizarán la métricas PSNR (Mean PSNR-Y) y SSIM (Mean SSIM-Y).

Para determinar la mejor opción dentro de los vídeos codificados se tendrán en cuenta el tamaño del bitstream obtenido y el valor de las métricas que evalúan la calidad. Cuanto menor sea el tamaño de bitstream y mejor sea el resultado proporcionado por la métrica respecto al vídeo codificado, mejor opción será éste. Nuestra intención es buscar un compromiso o trade-off de forma que una variable no predomine sobre la otra.

Representando en una gráfica los pares de valores (bitstream, métrica) de cada una de las codificaciones, aquellos valores que queden más a la izquierda (minimización de tamaño de bitstream) y arriba (maximización de la métrica) serán los óptimos en función de los requerimientos de este problema multiobjetivo, constituyendo lo que se conoce como frontera de Pareto. Si se utiliza la inversa de la métrica, tendremos estrictamente un problema de minimización y los puntos dominantes se encontrarán en la parte izquierda e inferior de la gráfica.

4.1 Caso de estudio1: PSNR

Para este caso de estudio se ha elegido la PSNR como métrica de la calidad. En las tablas 8-23 de esta sección se muestran los valores de bitstream y PSNR para los distintos valores de QP (15, 20, 25, 30, 35, 40, 45 y 50), crecientes ordinalmente en dichas tablas desde posiciones superiores a inferiores.

Las distintas combinaciones de filtros y thresholds en la etapa de preprocesado y el códec x265 con distintos QP's como parámetro producirán distintas salidas codificadas. Cada una de ellas será representada en una gráfica mediante su tamaño de bitstream y su valor de PSNR.

También se realizará la codificación del vídeo original para diferentes QP's, con objeto de obtener los bitstreams que serán comparados con los anteriores.

4.1.1 Experimentos 1 y 2

Se muestran en paralelo los resultados obtenidos para los dos vídeos de entrada (football_422_qcif.yuv y mother_daughter_qcif.yuv) en las distintas codificaciones realizadas.

4.1.1.1 Originales

Se muestran en las Tablas 8 y 9 los valores obtenidos después de llevar a cabo la codificación con el códec x265, fijando mediante él los ocho valores de QP indicados anteriormente y sin haber realizado ningún preprocesamiento. En las Figuras 31 y 32 aparecen representados en IVQC los distintos valores de PSNR obtenidos.

Bitstream	PSNR
5.805.632	47,08
4.188.621	42,46
2.905.735	37,88
1.883.192	33,55
1.166.176	29,76
743.773	26,65
521.561	24,14
427.215	22,22

Tabla 8. PSNR-Bitstream (football_original).

Bitstream	PSNR
3.014.201	47,48
2.143.141	43,23
1.477.745	39,20
977.600	35,46
649.064	32,09
452.365	29,08
340.898	26,21
280.942	23,77

Tabla 9. PSNR-Bitstream (mother_daughter_original).

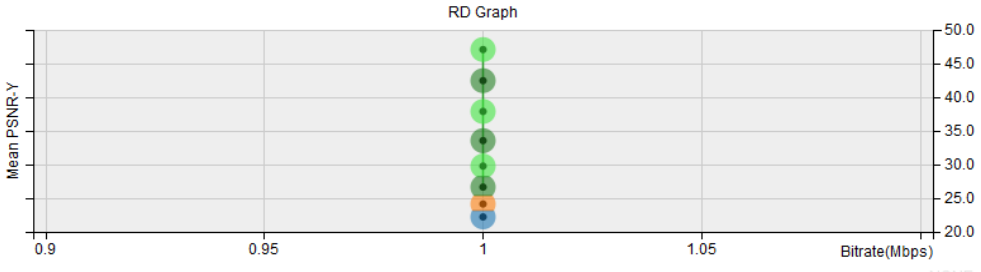

Figura 31. PSNR football_original (Intel Video Quality Caliper).

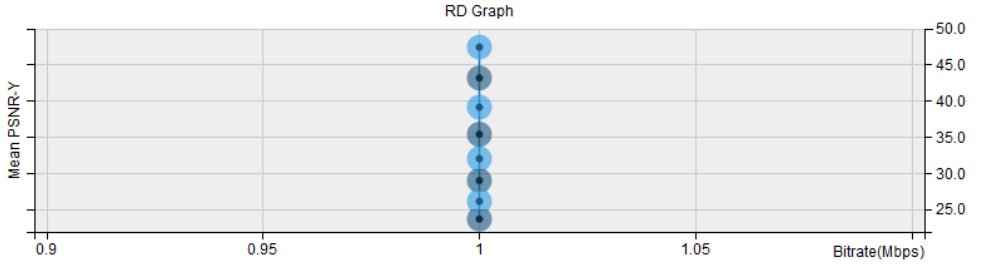

Figura 32. PSNR mother_daughter_original (Intel Video Quality Caliper).

4.1.1.2 Aplicación Mediana 3×3

En el este apartado sí se realiza preprocesado, para ello se aplica el filtro de mediana 3 × 3 que, como se explicó anteriormente, sustituye el valor de un pixel por aquel que ocupa el quinto lugar al ordenar de menor a mayor el conjunto formado por el propio pixel y sus ocho vecinos. Los resultados se muestran en las Tablas 10 y 11, y se representan los valores de PSNR gráficamente en las Figuras 33 y 34.

Bitstream	PSNR
4.255.211	27,08
2.912.278	27,02
1.933.988	26,85
1.238.492	26,48
823.369	25,75
603.437	24,70
484.256	23,41
420.892	22,02

Tabla 10. PSNR-Bitstream football_mediana3×3.

Bitstream	PSNR
2.567.781	35,38
1.780.261	34,98
1.221.935	34,13
836.825	32,67
587.283	30,66
431.040	28,42
333.926	25,95
280.011	23,74

Tabla 11. PSNR-Bitstream_mother_daughter_mediana3×3

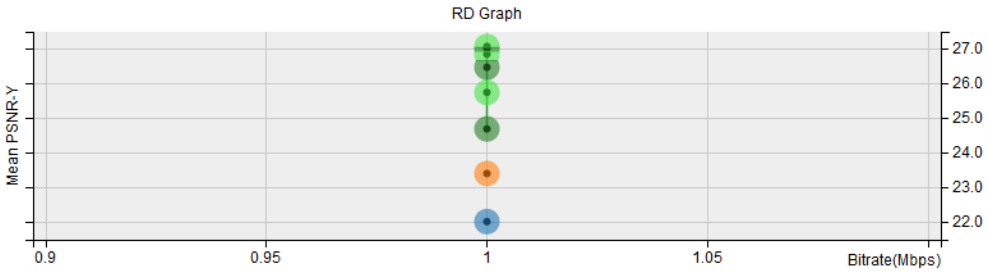

Figura 33. PSNR football_mediana3×3 (Intel Video Quality Caliper).

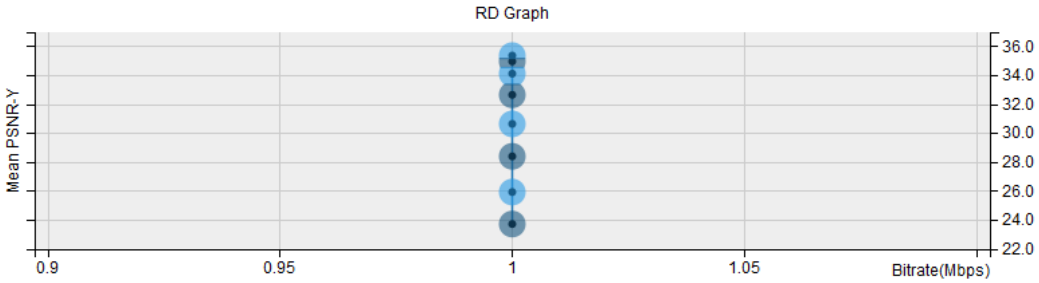

Figura 34. PSNR mother_daughter_mediana3×3 (Intel Video Quality Caliper).

4.1.1.3 Aplicación Mediana 5×5

El preprocesado de este apartado es similar al anterior, esta vez se tienen en cuenta 24 píxeles vecinos en vez de 8. El valor central del conjunto formado por 25 una vez ordenado, es el que tomará el pixel que se filtra en ese momento. Los valores obtenidos para este preprocesado se muestran en las Tablas 12 y 13.

Bitstream	PSNR
3.486.788	24,15
2.354.007	24,14
1.556.347	24,12
1.026.412	24,06
720.576	23,86
551.122	23,42
461.838	22,70
414.888	21,71

Tabla 12. PSNR-Bitstream football5×5.

Bitstream	PSNR
2.290.128	30,88
1.599.653	30,77
1.123.028	30,52
784.236	29,95
556.946	28,99
413.570	27,53
327.257	25,59
279.077	23,65

Tabla 13. PSNR-Bitstream mother_daughter5×5.

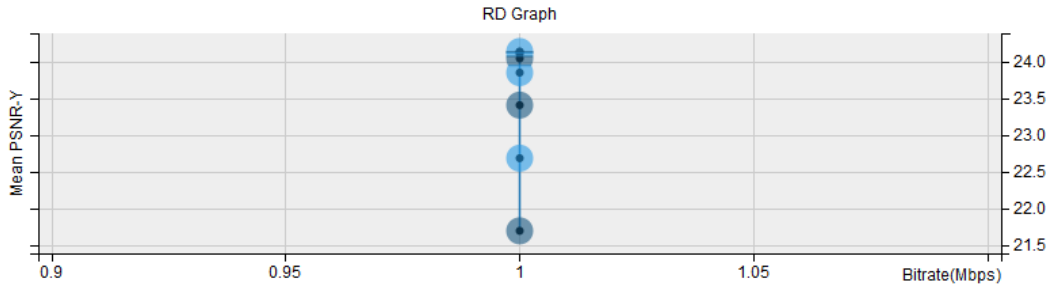

Figura 35. PSNR football_mediana5×5 (Intel Video Quality Caliper).

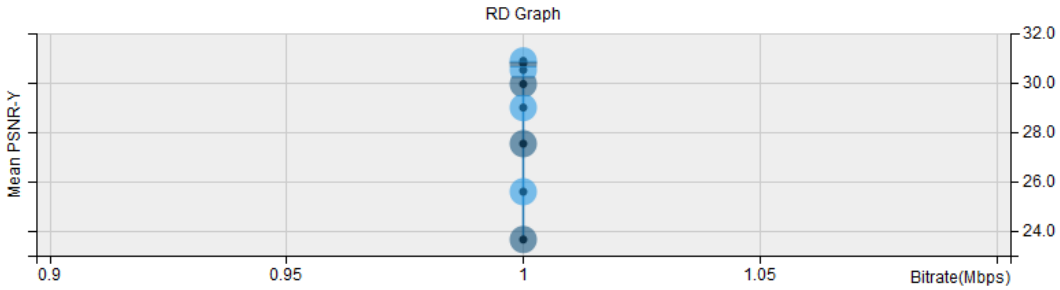

Figura 36. PSNR mother_daughter_mediana5×5 (Intel Video Quality Caliper).

4.1.1.4 Aplicación Combinado 3×3 (híbrido)

La combinación del filtro de mediana 3×3 y el operador de Sobel dará lugar al filtro que hemos denominado Combinado 3×3. El filtro y el operador se combinarán de manera que sólo se aplicará el filtro a aquellos píxeles que no se encuentran en alguno de los bordes de las figuras que integran el fotograma, preservando así los píxeles integrantes de bordes de dicho filtrado.

Como se ilustró anteriormente, para detectar dichos bordes se llevará a cabo el cálculo del gradiente, comparando después el resultado obtenido con un valor fijado previamente (threshold), se estimará si un pixel forma parte de un borde. Parametrizando la codificación con distintos valores de *threshold* se regulará el porcentaje de píxeles filtrados (píxeles no detectados como integrantes de bordes). A medida que dicho factor sea más pequeño, más píxeles serán reconocidos como integrantes de bordes y por tanto el filtrado efectivo será aplicado a un menor número de píxeles.

Threshold0

En las Tablas 14 y 15 y en las Figuras 37 y 38 se aprecian los resultados para un valor de threshold=0 y un filtro de mediana 3×3.

Bitstream	PSNR
4.886.366	30,85
3.502.058	30,71
2.453.349	30,30
1.629.600	29,37
1.037.445	27,77
682.154	25,76
503.941	23,83
424.639	22,15

Tabla 14 PSNR-Bitstream football_combt03×3.

Bitstream	PSNR
2.997.152	46,03
2.130.781	42,69
1.469.498	39,01
973.661	35,40
647.268	32,06
451.686	29,07
340.381	26,19
280.976	23,77

Tabla 15 PSNR-Bitstream mother_daughter_combt03×3.

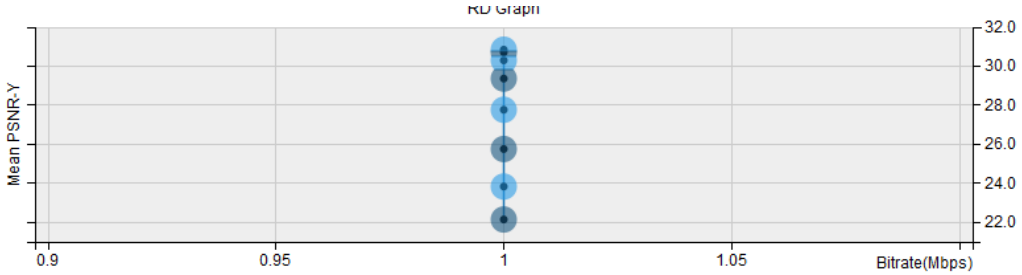

Figura 37. PSNR football_combt03×3 (Intel Video Quality Caliper).

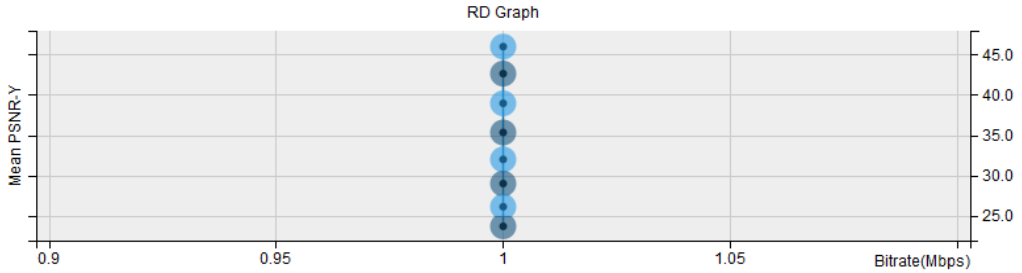

Figura 38. PSNR mother_daughter_combt03×3 (Intel Video Quality Caliper).

En la Tabla 16 y la Figura 39 se proporcionan los resultados para un valor de threshold=30 y un filtro de mediana 3×3 de la codificación que tiene como entrada el vídeo mother_daughter_420.yuv, mientras que en la Tabla 17 y la Figura 40 se aprecian los resultados para un valor de threshold=10 y un filtro de mediana 3×3. Los valores de los thresholds utilizados aquí ya se suponen buenas aproximaciones a los respectivos thresholds ideales.

Threshold30

Bitstream	PSNR
4.578.862	28,54
3.212.725	28,45
2.204.196	28,21
1.443.035	27,62
924.658	26,58
635.331	25,16
492.235	23,61
422.438	22,07

Tabla 16 PSNR-Bitstream football_combt303×3.

Threshold10

Bitstream	PSNR
2.755.740	37,91
1.945.229	37,24
1.334.118	35,94
893.033	33,87
612.004	31,37
440.077	28,80
336.642	26,08
280.351	23,76

Tabla 17 PSNR-Bitstream mother_daughter_combt103×3

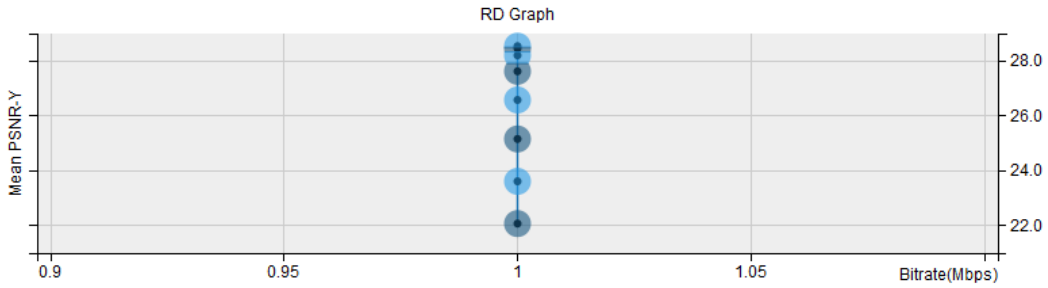

Figura 39. PSNR football_combt303×3 (Intel Video Quality Caliper).

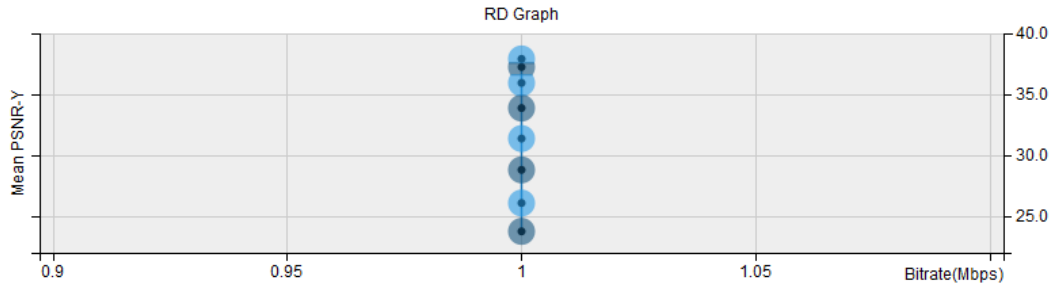

Figura 40. PSNR mother_daughter_combt103×3 (Intel Video Quality Caliper).

4.1.1.5 Aplicación de Combinado 5×5 (híbrido)

Es similar al apartado anterior, pero utiliza un filtro de mediana de 5×5 en vez de 3×3, se parte ahora de los últimos valores de threshold (30 y 10 respectivamente) utilizados en el Combinado 3×3. Los resultados obtenidos se muestran en las Tablas 18 y 19 y en las Figuras 41 y 42.

Threshold30

Bitstream	PSNR
5.065.314	32,26
3.679.230	32,07
2.574.417	31,57
1.703.099	30,43
1.081.475	28,49
705.575	26,13
508.903	23,92
424.736	22,15

Tabla 18 PSNR-Bitstream football_combt305×5

Threshold10

Bitstream	PSNR
2.973.252	42,63
2.107.135	40,93
1.443.736	38,29
957.119	35,12
640.551	31,95
449.611	29,03
339.876	26,20
280.897	23,76

Tabla 19 PSNR-Bitstream mother_daughter_combt105×5

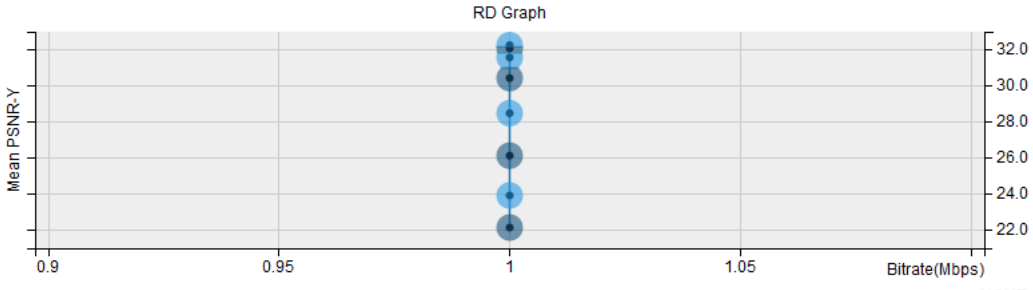

Figura 41. PSNR football_combt305×5 (Intel Video Quality Caliper).

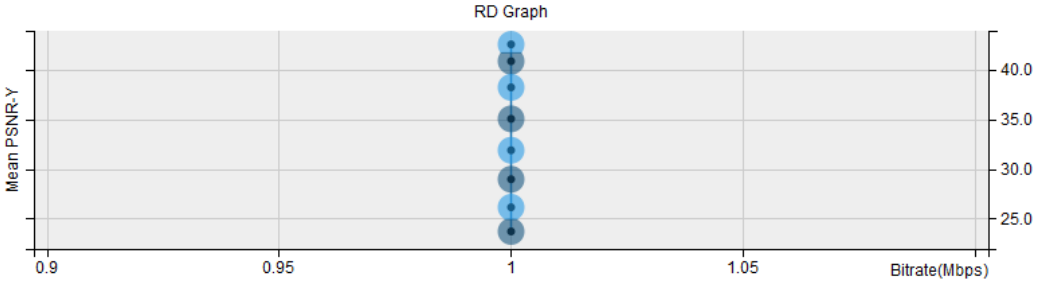

Figura 42 PSNR mother_daughter_combt105×5 (Intel Video Quality Caliper).

Se utilizan ahora los valores de threshold 40 y 30 respectivamente. Los resultados obtenidos se muestran en las tablas 20 y 21 y en las figuras 43 y 44.

Threshold40

Bitstream	PSNR
4.849.639	30,71
3.522.999	30,59
2.484.278	30,26
1.656.134	29,43
1.054.067	27,90
690.123	25,85
503.477	23,82
423.626	22,11

Tabla 20 PSNR-Bitstream football_combt405×5.

Threshold30

Bitstream	PSNR
2.764.607	36,51
1.968.038	36,11
1.351.015	35,21
898.429	33,56
611.405	31,24
438.794	28,75
337.109	26,08
280.165	23,75

Tabla 21 PSNR-Bitstream football_combt305×5.

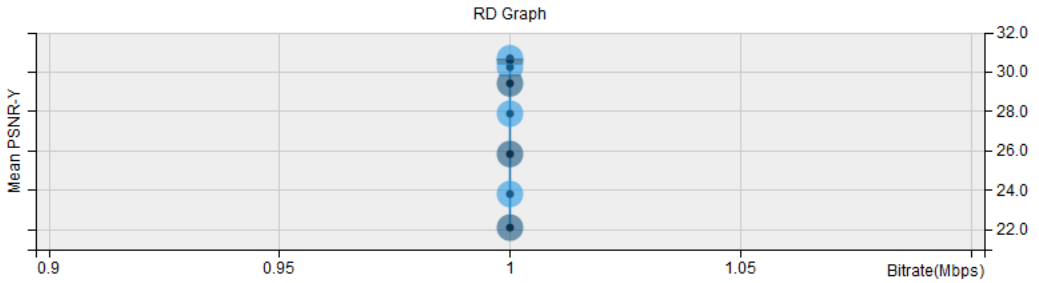

Figura 43. PSNR football_combt405×5 (Intel Video Quality Caliper).

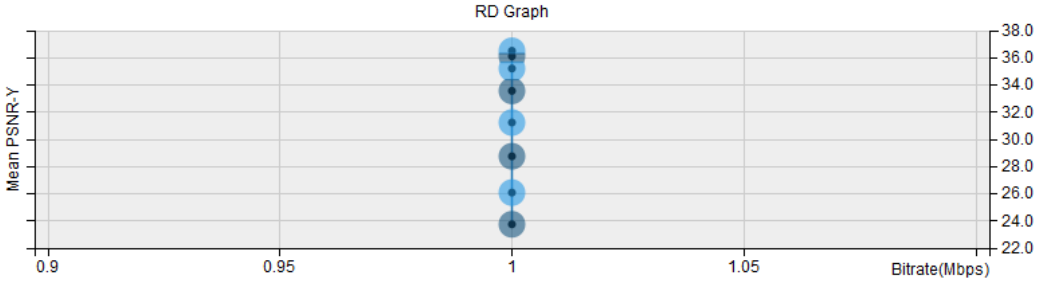

Figura 44. PSNR mother_daughter_combt305×5 (Intel Video Quality Caliper).

Se tienen ahora los valores máximos de threshold tomados en cada uno de los experimentos (60 y 40 respectivamente). Los resultados obtenidos se muestran en las Tablas 22 y 23 y en las Figuras 45 y 46.

Threshold60

Bitstream	PSNR
4.575.227	28,64
3.311.318	28,57
2.337.602	28,38
1.562.123	27,87
997.249	26,83
658.300	25,27
493.321	23,59
421.517	22,04

Threshold40

Bitstream	PSNR
2.670.028	35,02
1.896.236	34,75
1.307.014	34,11
872.442	32,84
598.086	30,85
433.836	28,58
334.901	26,01
279.814	23,73

Tabla 22 PSNR-Bitstream football_combt605×5. Tabla 23 PSNR-Bitstream mother_daughter _combt405×5.

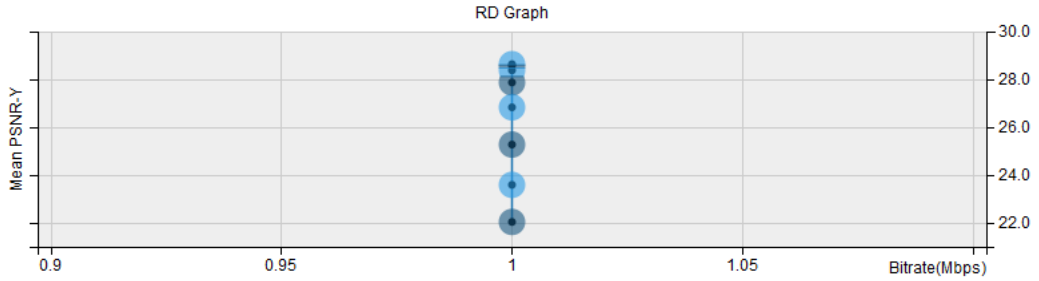

Figura 45 PSNR football_combt605×5 (Intel Video Quality Caliper).

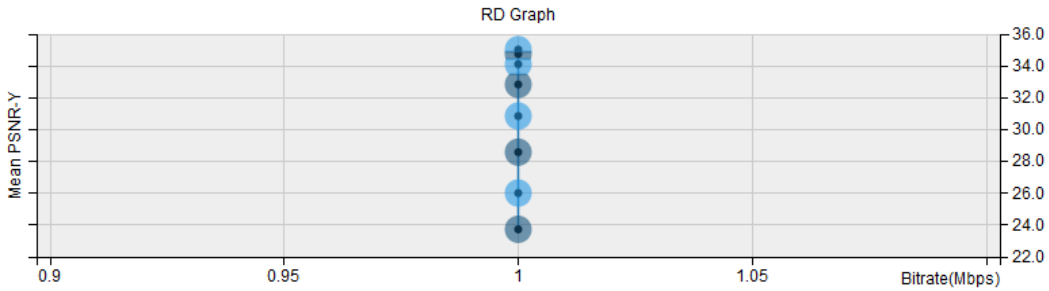

Figura 46 PSNR mother_daughtert405×5 (Intel Video Quality Caliper).

4.1.2 Pareto experimento 1

Para llevar a cabo el cálculo de las configuraciones dominantes en el primer vídeo tomado como estímulo (football) se han realizado dos gráficas a partir de los valores de tamaño del bitstream y de inversa de la PSNR de cada una de las codificaciones. Se trata, en este caso, de un problema de minimización y la frontera de Pareto recorrerá la parte inferior e izquierda de las gráficas. En la Figura 47 se muestra la primera gráfica, en la que aparecen representados todos los puntos determinados por los pares (tamaño bitstream, 1/PSNR).

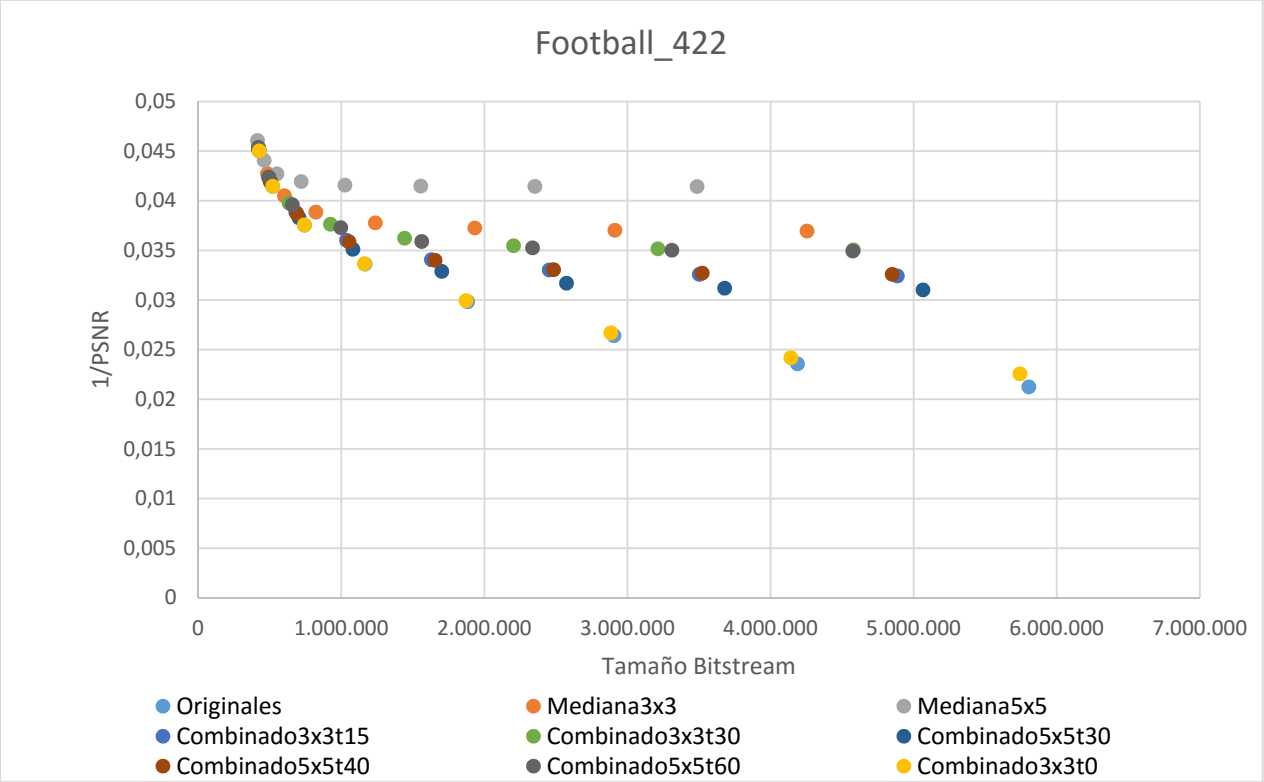

Figura 47. Posibles soluciones para mejor trade-off Bitstream-PSNR (football).

En la Figura 48 se resaltan en color rojo los puntos que constituyen la frontera de Pareto. Tal y como se expuso anteriormente, éstos forman parte de la parte izquierda e inferior de la gráfica.

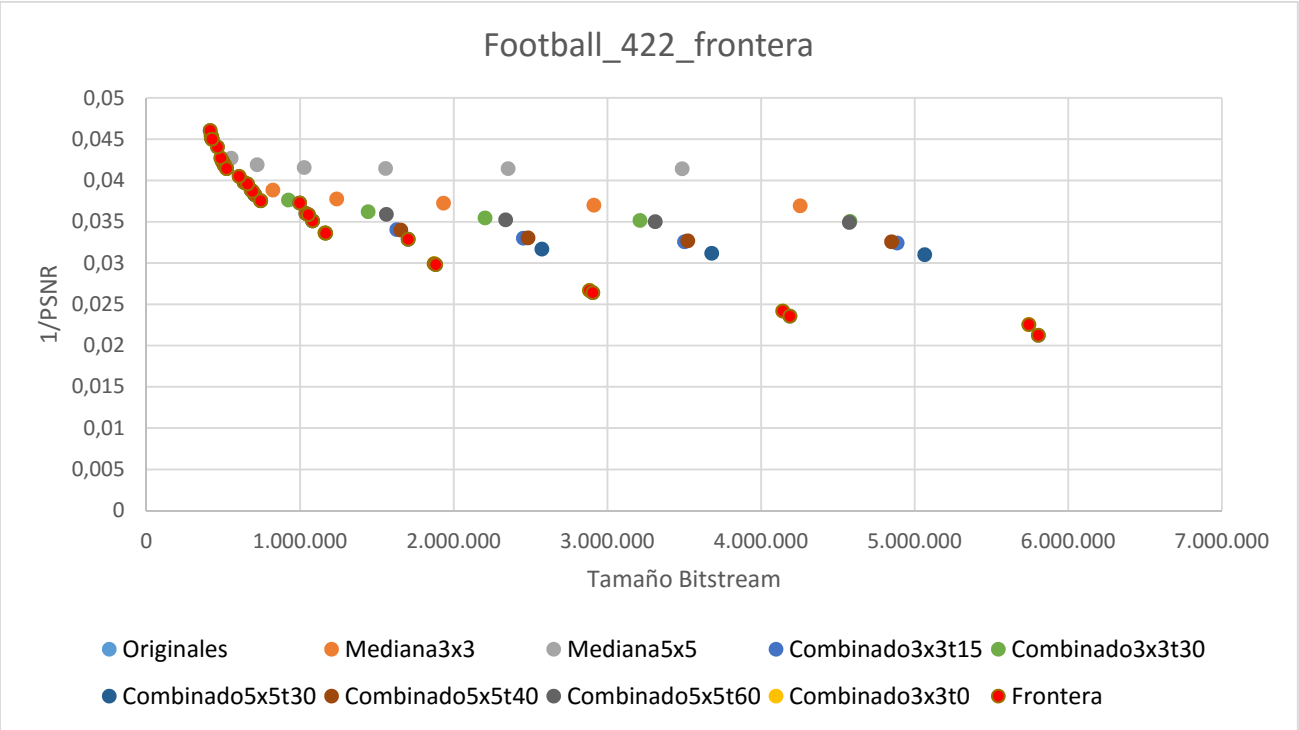

Figura 48. Frontera de Pareto (football).

La mayoría de los puntos que constituyen la frontera corresponden a codificaciones de vídeos originales o a decodificaciones realizadas posteriormente a la aplicación de filtros combinados. Estas últimas integran en el mejor de los casos cuatro puntos en la frontera de Pareto y dichos puntos tienen que ver con codificaciones en las que se utilizaron valores altos de QP (35-50), dichos valores de QP se corresponden con bitrates bajas. Sólo cinco puntos pertenecen a codificaciones posteriores a la aplicación de los filtros de mediana y estas codificaciones se realizaron con valores de QP que oscilan entre 40 y 50.

4.1.3 Pareto experimento 2

Al igual que en el experimento anterior se han representado dos gráficas que se pueden apreciar en las Figuras 49 y 50. En la segunda vuelven a representarse en rojo los puntos que pertenecen a la zona de dominancia de la gráfica.

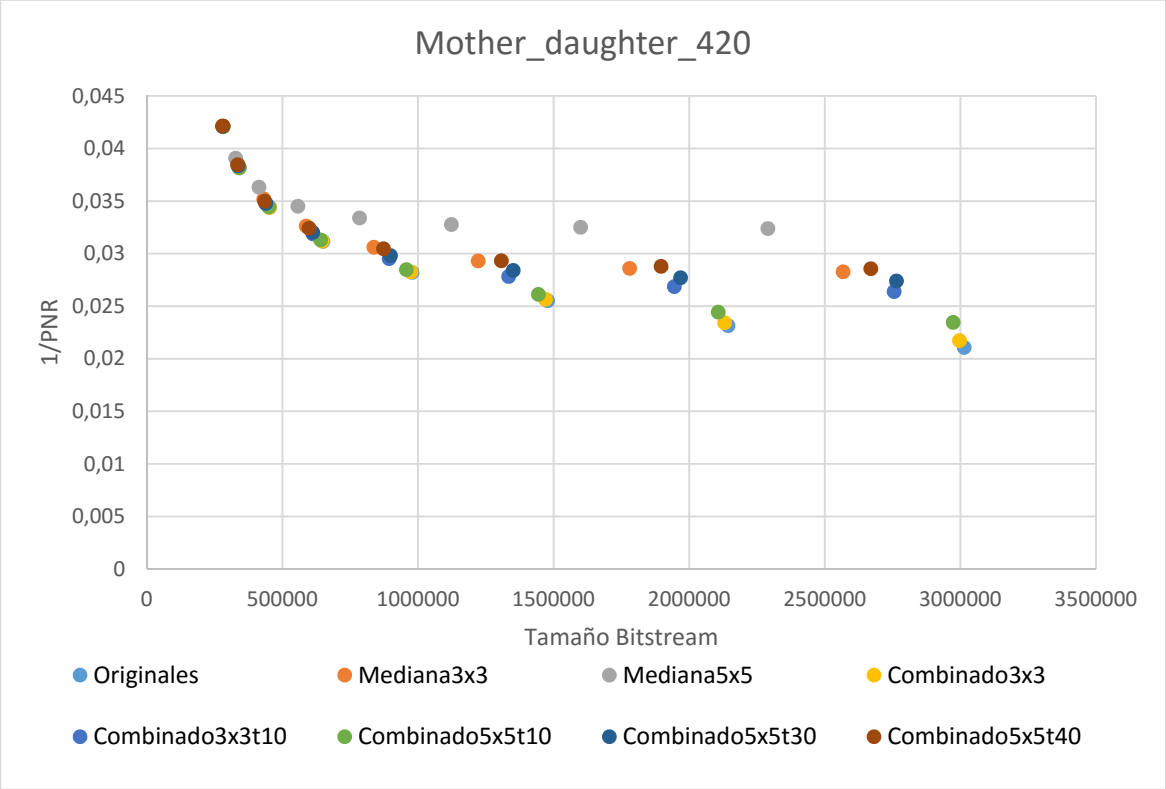

Figura 49. Posibles soluciones para mejor trade-off Bitstream-PSNR (mother_daughter).

Al igual que en el experimento anterior la mayoría de los puntos de la zona de dominancia corresponden a codificaciones de vídeos originales o a codificaciones posteriores a la aplicación de filtros combinados. En este caso son ocho los puntos que pertenecen a codificaciones posteriores a la aplicación de los filtros de mediana, entre los valores de QP utilizados para estas codificaciones se encuentra como valor más bajo el 30. Igualmente para las opciones de filtros combinados se tienen puntos de dominancia con QP's inferiores a 35, en particular se tiene una codificación con QP igual a 20, siendo el threshold igual a 10 y el filtro aplicado el de mediana 5×5.

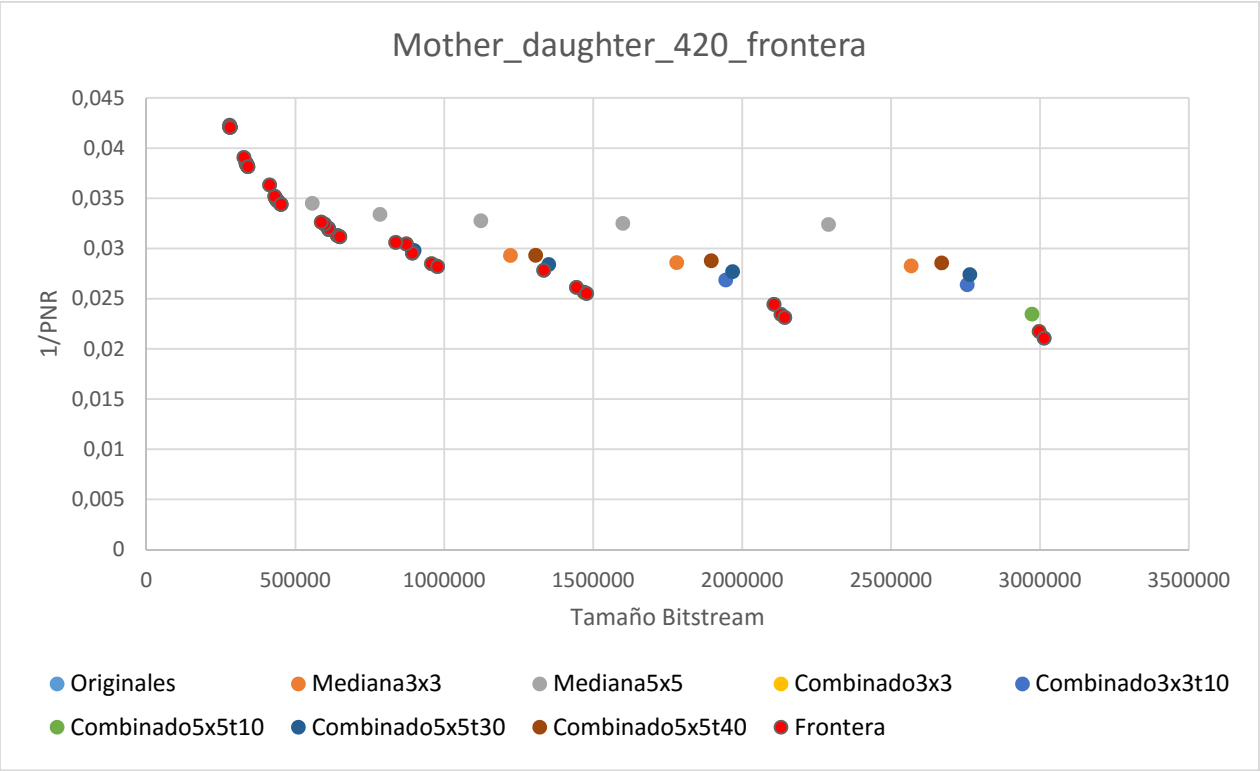

Figura 50. Frontera de Pareto (mother_daughter).

El preprocesado ha dado mejores resultados en el segundo vídeo que en el primero, apareciendo codificaciones preprocesadas con QP's bajas en la zona de dominancia. A pesar de este fenómeno, las codificaciones que mejores resultados aportan son aquellas que combinan filtro de mediana y operador de Sobel y para las que se tienen valores de threshold muy bajos, es decir aquellas que se asemejan más a las codificaciones del vídeo original. Como cabía esperar el filtrado reduce el tamaño de bitstream, sin embargo también reduce la calidad cuando se utiliza la métrica PSNR para estimarla.

Se muestra en la Tabla 24 un resumen de aquellas combinaciones con preprocesado que aparecen en la frontera de Pareto de cada uno de los experimentos y cuyos valores de thresholds están más próximos al ideal (por tanto más alejadas de las codificaciones originales). En la Tabla 24 también se muestran las codificaciones que no hacen uso del preprocesado (originales) correspondiéndose a las primeras (en lo que a QP se refiere).

Combinación	Bitstream	1/PSNR
Experimento1		
Originales. QP =30	1883192	0,030
Originales QP = 35	1166176	0,034
Mediana 3×3, threshold = 15 y QP = 35	1037445	0,036
Mediana 3×3, threshold = 15 y QP = 30	1703099	0,032
Mediana 5×5, threshold = 40 y QP = 35	1054067	0,036
Mediana 5×5, threshold = 60 y QP = 35	997249	0,037
Experimento2		
Originales QP = 25	1477745	0,025
Mediana 3×3, threshold = 10 y QP = 25	1334118	0,028
Mediana 5×5, threshold = 10 y QP = 25	2107135	0,024
Mediana 3×3	836825	0,031

Tabla 24. Resultados de las combinaciones más concluyentes de los experimentos uno y dos.

4.2 Caso de estudio 2: SSIM

Hemos remarcado previamente que la métrica PSNR está basada en el dominio de procesamiento de señal y puede que los valores que arroja no guarden relación con la percepción de un individuo al visualizar un vídeo. La métrica SSIM proporciona valores que se asemejan más a la percepción que tendría el ojo humano, por tanto es más adecuada para el estudio que se lleva a cabo en este trabajo.

Para el cálculo de la frontera de Pareto, se ha utilizado el valor del tamaño de bitstream en el eje *x* y la métrica SSIM en el eje *y*, por lo que en el eje *x* se busca minimización y en el eje *y* la maximización. La frontera se compondrá por tanto de puntos en la parte alta de la gráfica y de puntos en la parte izquierda de dicha gráfica.

4.2.1 Experimento 3

En la Tabla 25 se muestran los nombres de los bitstreams obtenidos en las diferentes codificaciones del vídeo mother_daughter_420, en dichos nombres figura el QP utilizado en la

codificación así como el threshold (en los casos en los que se haya utilizado) y el tipo de preprocesado cuando lo haya habido. Para cada bitstream se detalla su tamaño y el valor de la métrica SSIM (que toma valores entre 0 y 1) ofrecido por la herramienta Intel Video Quality Caliper.

Nombre	Bitstream(bytes)	SSIM
Mediana 3×3 + Operador de Sobel con threshold = 0		
mother_daughter_420_x265_QP15comb.hevc	2997152	0,99
mother_daughter_420_x265_QP20_comb.hevc	2130781	0,99
mother_daughter_420_x265_QP25comb.hevc	1469498	0,96
mother_daughter_420_x265_QP30comb.hevc	973661	0,91
mother_daughter_420_x265_QP35_comb.hevc	647268	0,83
mother_daughter_420_x265_QP40comb.hevc	451686	0,73
mother_daughter_420_x265_QP45_comb.hevc	340381	0,62
mother_daughter_420_x265_QP50comb.hevc	280976	0,53
Mediana 3×3 + Operador de Sobel con threshold = 10		
mother_daughter_420x265_QP15comb3×3t10.hevc	2755740	0,96
mother_daughter_420x265_QP20_comb3×3t10.hevc	1945229	0,95
mother_daughter_420x265_QP25comb3×3t10.hevc	1334118	0,92
mother_daughter_420x265_QP30comb3×3t10.hevc	893033	0,88
mother_daughter_420x265_QP35_comb3×3t10.hevc	612004	0,81
mother_daughter_420x265_QP40comb3×3t10.hevc	440077	0,72
mother_daughter_420x265_QP45_comb3×3t10.hevc	336642	0,61
mother_daughter_420x265_QP50comb3×3t10.hevc	280351	0,54
Mediana 5×5 + Operador de Sobel con threshold = 10		
mother_daughter_420x265_QP15comb5×5t10.hevc	2973252	0,98
mother_daughter_420x265_QP20_comb5×5t10.hevc	2107135	0,97
mother_daughter_420x265_QP25comb5×5t10.hevc	1443736	0,95
mother_daughter_420x265_QP30comb5×5t10.hevc	957119	0,9
mother_daughter_420x265_QP35_comb5×5t10.hevc	640551	0,83
mother_daughter_420x265_QP40comb5×5t10.hevc	449611	0,73
mother_daughter_420x265_QP45_comb5×5t10.hevc	339876	0,62
mother_daughter_420x265_QP50comb5×5t10.hevc	280897	0,54
Mediana 5×5 + Operador de Sobel con threshold = 30		
mother_daughter_420x265_QP15comb5×5t30.hevc	2764607	0,93
mother_daughter_420x265_QP20_comb5×5t30.hevc	1968038	0,92
mother_daughter_420x265_QP25comb5×5t30.hevc	1351015	0,9
mother_daughter_420x265_QP30comb5×5t30.hevc	898429	0,87
mother_daughter_420x265_QP35_comb5×5t30.hevc	611405	0,8
mother_daughter_420x265_QP40comb5×5t30.hevc	438794	0,71
mother_daughter_420x265_QP45_comb5×5t30.hevc	337109	0,61
mother_daughter_420x265_QP50comb5×5t30.hevc	280165	0,54

Nombre	Bitstream(bytes)	SSIM
Mediana 5×5 + Operador de Sobel con threshold = 40		
mother_daughter_420x265_QP15comb5×5t40.hevc	2670028	0,91
mother_daughter_420x265_QP20_comb5×5t40.hevc	1896236	0,9
mother_daughter_420x265_QP25comb5×5t40.hevc	1307014	0,89
mother_daughter_420x265_QP30comb5×5t40.hevc	872442	0,85
mother_daughter_420x265_QP35_comb5×5t40.hevc	598086	0,79
mother_daughter_420x265_QP40comb5×5t40.hevc	433836	0,7
mother_daughter_420x265_QP45_comb5×5t40.hevc	334901	0,6
mother_daughter_420x265_QP50comb5×5t40.hevc	279814	0,54
Mediana 3×3		
mother_daughter_420_x265_QP15med.hevc	2567781	0,94
mother_daughter_420_x265_QP20_med.hevc	1780261	0,93
mother_daughter_420_x265_QP25med.hevc	1221935	0,91
mother_daughter_420_x265_QP30med.hevc	836825	0,87
mother_daughter_420_x265_QP35_med.hevc	587283	0,8
mother_daughter_420_x265_QP40med.hevc	431040	0,71
mother_daughter_420_x265_QP45_med.hevc	333926	0,61
mother_daughter_420_x265_QP50med.hevc	280011	0,54
Mediana 5×5		
mother_daughter_420_qcif_x265_QP15med5×5.hevc	2290128	0,85
mother_daughter_420_qcif_x265_QP20_med5×5.hevc	1599653	0,85
mother_daughter_420_qcif_x265_QP25med5×5.hevc	1123028	0,83
mother_daughter_420_qcif_x265_QP30med5×5.hevc	784236	0,8
mother_daughter_420_qcif_x265_QP35_med5×5.hevc	556946	0,75
mother_daughter_420_qcif_x265_QP40med5×5.hevc	413570	0,67
mother_daughter_420_qcif_x265_QP45_med5×5.hevc	327257	0,59
mother_daughter_420_qcif_x265_QP50med5×5.hevc	279077	0,53
Originales		
mother_daughter_420_x265_QP15.hevc	3014201	0,99
mother_daughter_420_x265_QP20.hevc	2143141	0,98
mother_daughter_420_x265_QP25.hevc	1477745	0,96
mother_daughter_420_x265_QP30.hevc	977600	0,91
mother_daughter_420_x265_QP35.hevc	649064	0,84
mother_daughter_420_x265_QP40.hevc	452365	0,73
mother_daughter_420_x265_QP45.hevc	340898	0,62
mother_daughter_420_x265_QP50.hevc	280942	0,54

Tabla 25. Bitstreams y SSIM para distintos valores de QP y Preprocesados (mother_daughter).

4.2.1.1 Originales

Como sucedía en el anterior caso de estudio, los valores de la métrica (SSIM en este caso) para las codificaciones sin preprocesado (originales) son difícilmente mejorables. Especialmente aquellas codificaciones con un QP bajo (valor alto de bitstream). Los valores proporcionados por la herramienta para estas codificaciones se ilustran en la Figura 51.

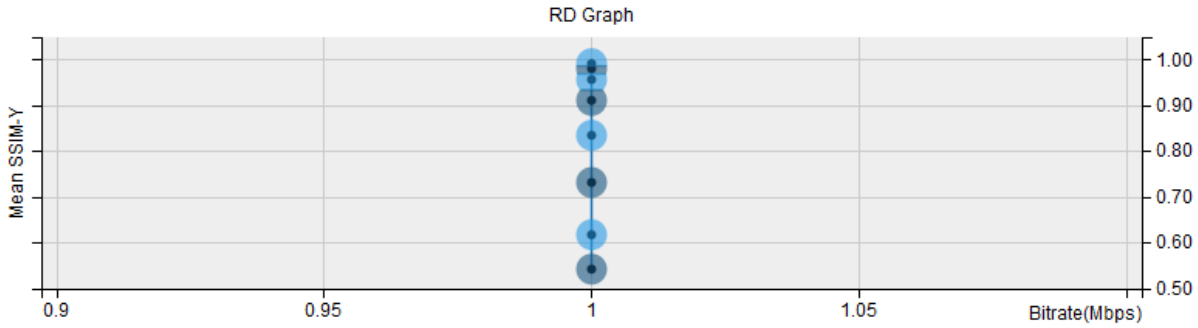

Figura 51. SSIM mother_daughter_orig (Intel Video Quality Caliper).

4.2.1.2 Aplicación de Mediana 3×3

En la Figura 52 se puede apreciar una gran semejanza entre la codificación sin preprocesado y la codificación que utiliza el filtro de mediana 3×3 para preprocesar en cuanto a los valores obtenidos para la estimación de la calidad.

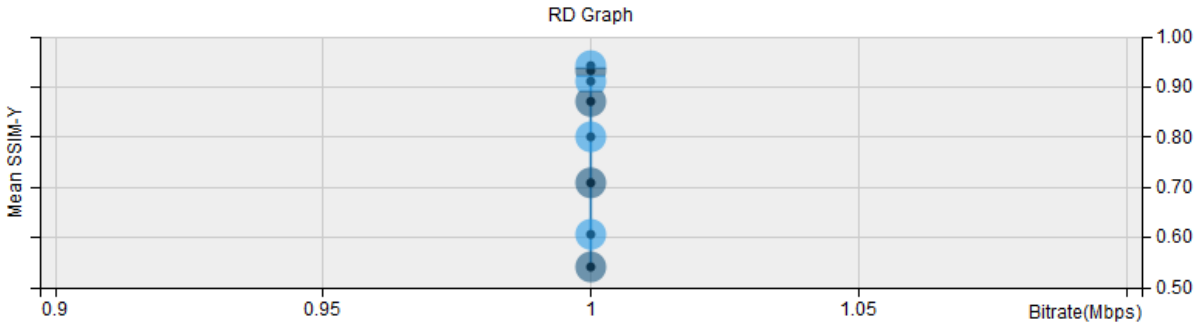

Figura 52. SSIM mother_daughter_med3×3 (Intel Video Quality Caliper).

4.2.1.3 Aplicación de Mediana 5×5

Los valores para esta variedad del filtro de mediana descienden sensiblemente si los comparamos con los de la mediana 3×3, siendo este descenso más acusado en los valores más altos de QP. Como contrapartida, este filtrado obtiene resultados mejores que el anterior en lo que a tamaño de bitstream se refiere. Los valores de SSIM obtenidos en IVQC se muestran en la Figura 53.

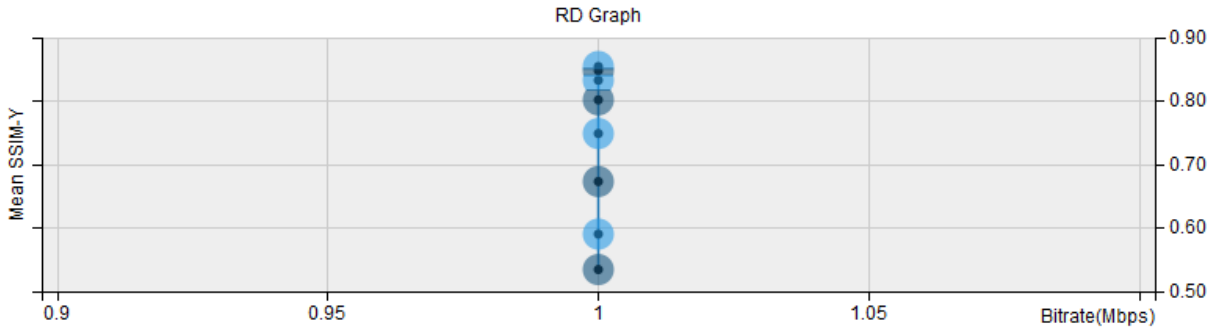

Figura 53. SSIM mother_daughter_med5×5 (Intel Video Quality Caliper).

4.2.1.4 Aplicación Combinado 3×3 (híbrido)

Threshold0

En la Figura 54 se puede observar como parametrizando el preprocesamiento con este valor de threshold se obtienen valores de SSIM muy similares a los de las codificaciones sin preprocesamiento, sin embargo apenas se reduce la compresión en relación con dichas codificaciones. Este resultado era esperable pues con este valor de threshold el filtro de mediana se aplica a una tasa de píxeles bastante baja.

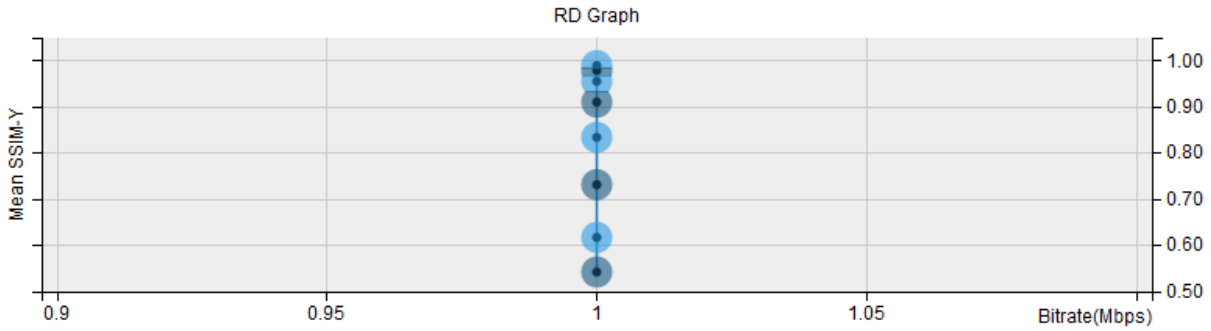

Figura 54. SSIM mother_daughtert03×3 (Intel Video Quality Caliper).

Threshold10

Para este valor de threshold se obtienen valores de SSIM (Figura 55) que se aproximan a los anteriores incluso para valores de QP bajos. Además se logra una tasa de compresión mayor.

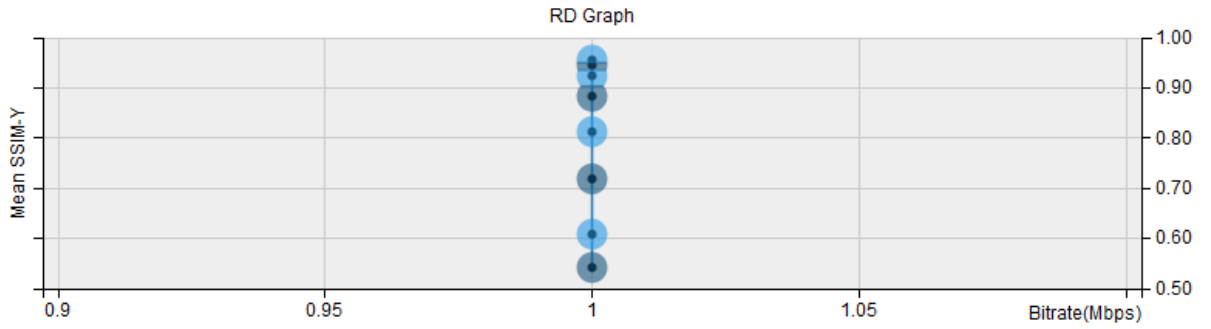

Figura 55. SSIM mother_daughtert103×3 (Intel Video Quality Caliper).

4.2.1.5 Aplicación de Combinado 5×5 (híbrido)

Threshold10

Al igual que sucedía para la combinación mediana 3×3 y threshold 0, los resultados recopilados para la combinación mediana 5×5 y threshold=10 (Figura 56) se aproximan mucho a los obtenidos en las codificaciones sin preprocesado. Los valores de tamaño de bitstream de estas dos combinaciones son también muy similares.

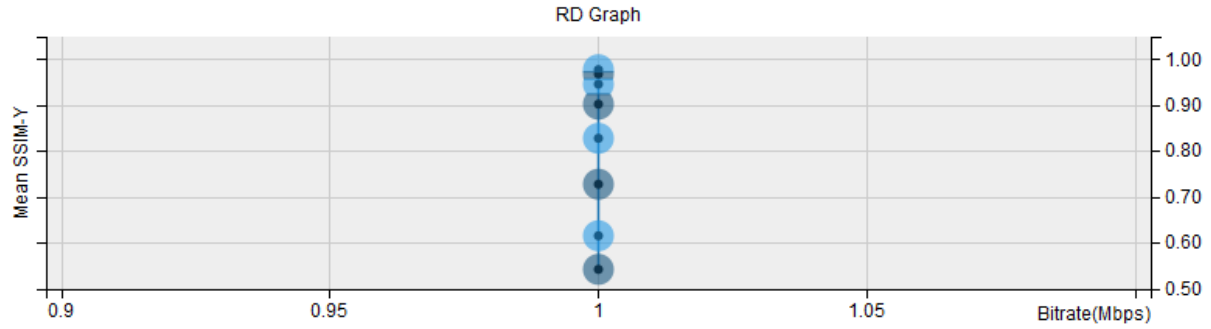

Figura 56. SSIM mother_daughtert105×5 (Intel Video Quality Caliper).

Threshold30

Los valores que se obtienen, tanto de SSIM (Figura 57) como de tamaño de bitstream, se acercan mucho a los obtenidos para la combinación mediana 3×3 threshold igual a 10.

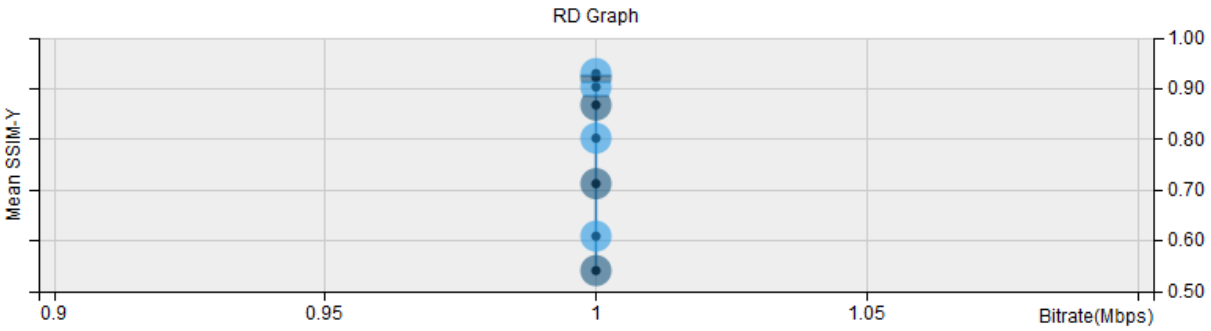

Figura 57. SSIM mother_daughtert305×5 (Intel Video Quality Caliper).

Theshold40

Para este valor de threshold los valores de SSIM (Figura 58) descienden en relación con los de los dos valores de thresholds vistos anteriormente, sin embargo éstos siguen siendo altos. Además esta combinación ofrece una tasa de compresión superior a la de las dos combinaciones anteriores.

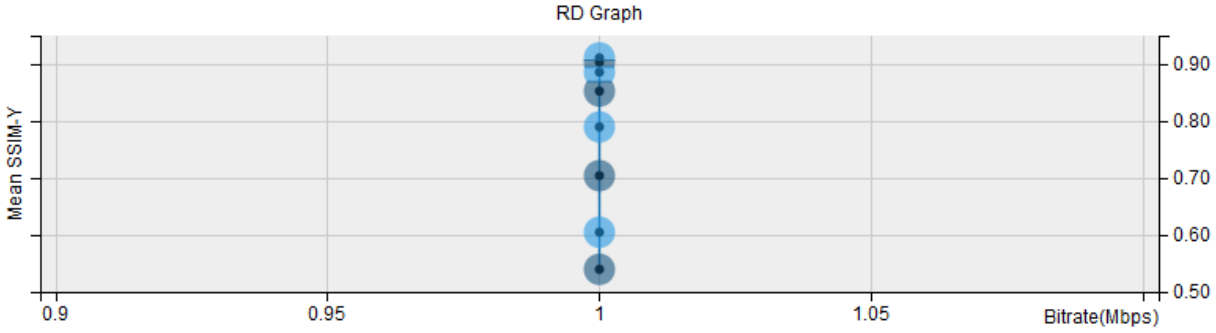

Figura 58. SSIM mother_daughtert405×5 (Intel Video Quality Caliper).

4.2.2 Experimento 4

En la Tabla 26 aparecen los nombres de los bitstreams obtenidos en las diferentes codificaciones del vídeo football_422, así como sus tamaños y los valores de la métrica SSIM medidos por la herramienta Video Quality Caliper. Como ocurría en el experimento anterior en los nombres se detallan QP, threshold y preprocesado.

Nombre	Bitstream(bytes)	SSIM
Mediana 3×3 + Operador de Sobel con threshold = 15		
football_422x265_QP15comb3×3t15.hevc	4886366	0,87
football_422x265_QP20comb3×3t15.hevc	3502058	0,87
football_422x265_QP25comb3×3t15.hevc	2453349	0,85
football_422x265_QP30comb3×3t15.hevc	1629600	0,81
football_422x265_QP35comb3×3t15.hevc	1037445	0,76
football_422x265_QP40comb3×3t15.hevc	682154	0,69
football_422x265_QP45_comb3×3t15.hevc	503941	0,6
football_422x265_QP50comb3×3t15.hevc	424639	0,51
Mediana 3×3 + Operador de Sobel con threshold = 30		
football_422x265_QP15comb3×3t30.hevc	4578862	0,84
football_422x265_QP20_comb3×3t30.hevc	3212725	0,83
football_422x265_QP25comb3×3t30.hevc	2204196	0,82
football_422x265_QP30comb3×3t30.hevc	1443035	0,78
football_422x265_QP35_comb3×3t30.hevc	924658	0,73
football_422x265_QP40comb3×3t30.hevc	635331	0,67
football_422x265_QP45_comb3×3t30.hevc	492235	0,59
football_422x265_QP50comb3×3t30.hevc	422438	0,51

Nombre	Bitstream(bytes)	SSIM
Mediana 5×5 + Operador de Sobel con threshold = 30		
football_422x265_QP15comb5×5t30.hevc	5065314	0,86
football_422x265_QP20_comb5×5t30.hevc	3679230	0,86
football_422x265_QP25comb5×5t30.hevc	2574417	0,84
football_422x265_QP30comb5×5t30.hevc	1703099	0,82
football_422x265_QP35_comb5×5t30.hevc	1081475	0,77
football_422x265_QP40comb5×5t30.hevc	705575	0,7
football_422x265_QP45_comb5×5t30.hevc	508903	0,61
football_422x265_QP50comb5×5t30.hevc	424736	0,51
Mediana 5×5 + Operador de Sobel con threshold = 40		
football_422x265_QP15comb5×5t40.hevc	4849639	0,83
football_422x265_QP20_comb5×5t40.hevc	3522999	0,83
football_422x265_QP25comb5×5t40.hevc	2484278	0,82
football_422x265_QP30comb5×5t40.hevc	1656134	0,79
football_422x265_QP35_comb5×5t40.hevc	1054067	0,75
football_422x265_QP40comb5×5t40.hevc	690123	0,69
football_422x265_QP45_comb5×5t40.hevc	503477	0,6
football_422x265_QP50comb5×5t40.hevc	423626	0,51
Mediana 5×5 + Operador de Sobel con threshold = 60		
football_422x265_QP15comb5×5t60.hevc	4575227	0,79
football_422x265_QP20_comb5×5t60.hevc	3311318	0,79
football_422x265_QP25comb5×5t60.hevc	2337602	0,78
football_422x265_QP30comb5×5t60.hevc	1562123	0,76
football_422x265_QP35_comb5×5t60.hevc	997249	0,72
football_422x265_QP40comb5×5t60.hevc	658300	0,66
football_422x265_QP45_comb5×5t60.hevc	493321	0,59
football_422x265_QP50comb5×5t60.hevc	421517	0,5
Mediana 3×3		
football_422_qcif_x265_QP15med.hevc	4255211	0,82
football_422_qcif_x265_QP20_med.hevc	2912278	0,81
football_422_qcif_x265_QP25med.hevc	1933988	0,8
football_422_qcif_x265_QP30med.hevc	1238492	0,77
football_422_qcif_x265_QP35_med.hevc	823369	0,72
football_422_qcif_x265_QP40med.hevc	603437	0,66
football_422_qcif_x265_QP45_med.hevc	484256	0,59
football_422_qcif_x265_QP50med.hevc	420892	0,51

Nombre	Bitstream(bytes)	SSIM
Mediana 5×5		
football_422_qcif_x265_QP15med5×5.hevc	3486788	0,67
football_422_qcif_x265_QP20_med5×5.hevc	2354007	0,67
football_422_qcif_x265_QP25med5×5.hevc	1556347	0,66
football_422_qcif_x265_QP30med5×5.hevc	1026412	0,65
football_422_qcif_x265_QP35_med5×5.hevc	720576	0,63
football_422_qcif_x265_QP40med5×5.hevc	551122	0,6
football_422_qcif_x265_QP45_med5×5.hevc	461838	0,55
football_422_qcif_x265_QP50med5×5.hevc	414888	0,49
Originales		
football_422_qcif_x265_QP15.hevc	5805632	0,99
football_422_qcif_x265_QP20.hevc	4188621	0,99
football_422_qcif_x265_QP25.hevc	2905735	0,96
football_422_qcif_x265_QP30.hevc	1883192	0,91
football_422_qcif_x265_QP35.hevc	1166176	0,83
football_422_qcif_x265_QP40.hevc	743773	0,73
football_422_qcif_x265_QP45.hevc	521561	0,62
football_422_qcif_x265_QP50.hevc	427215	0,52

Tabla 26. Bitstreams y SSIM para distintos valores de QP y Preprocesados (football).

4.2.2.1 Originales

Se muestran en la Figura 59 los valores de SSIM para las codificaciones sin preprocesado del vídeo football_422_qcif.yuv.

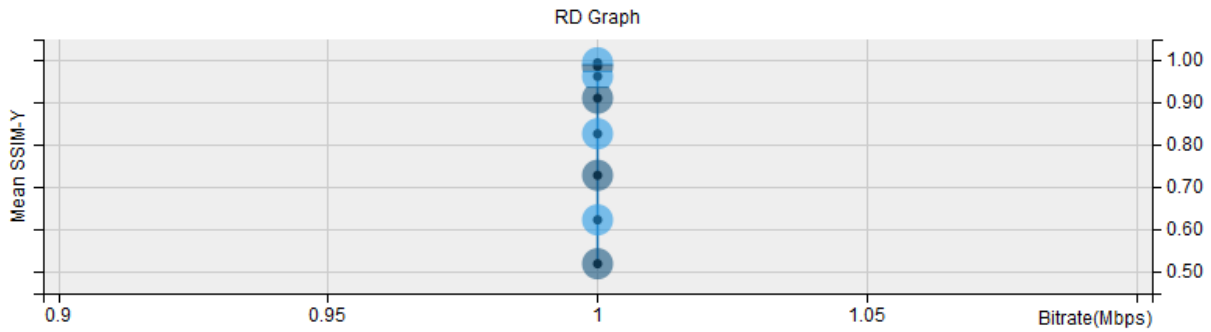

Figura 59. SSIM football_orig (Intel Video Quality Caliper).

4.2.2.2 Aplicación Mediana3×3

En la Figura 60 se observa que a diferencia de lo que ocurría en el experimento tres, los valores de SSIM para este tipo de preprocesado no alcanzan valores próximos a los de las codificaciones sin la etapa de preprocesado.

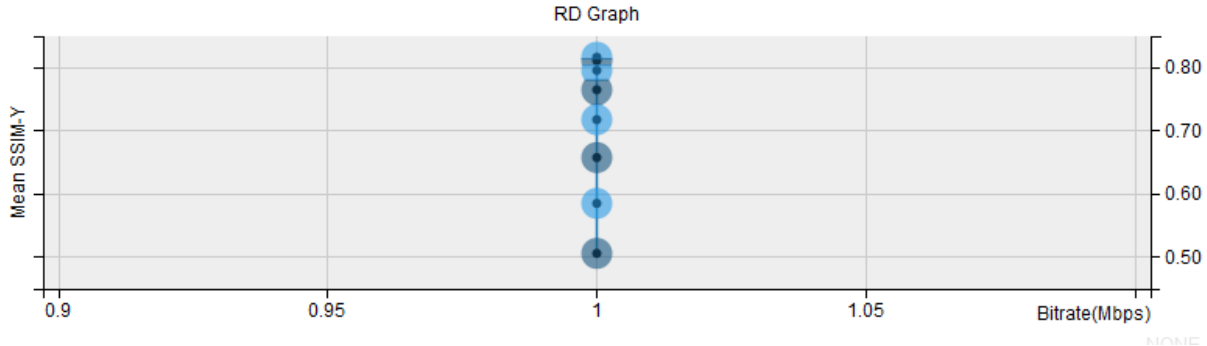

Figura 60. SSIM football_med3×3 (Intel Video Quality Caliper).

4.2.2.3 Aplicación Mediana 5×5

Como ocurría en el experimento anterior, se obtienen peores resultados para este filtrado de los que se obtenían para el filtrado de mediana 3×3 en lo que a la métrica SSIM se refiere. Los resultados obtenidos para el filtrado 5×5 se muestran en la Figura 61.

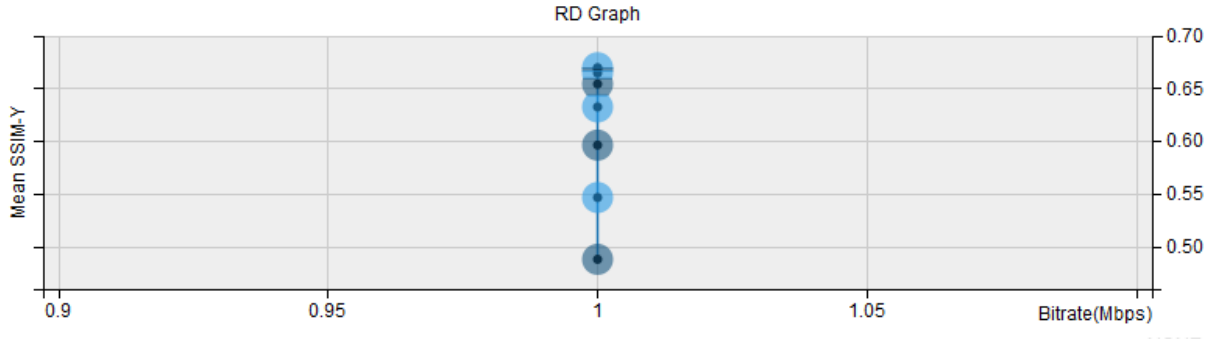

Figura 61. SSIM football_med5×5 (Intel Video Quality Caliper).

4.2.2.4 Aplicación Combinado3×3 (híbrido)

Threshold0

Tanto en este experimento como en el anterior, en esta combinación (Mediana 3×3 y threshold igual a 0) se obtienen valores de SSIM (Figura 62) muy próximos a los obtenidos para la codificación sin etapa de preprocesado.

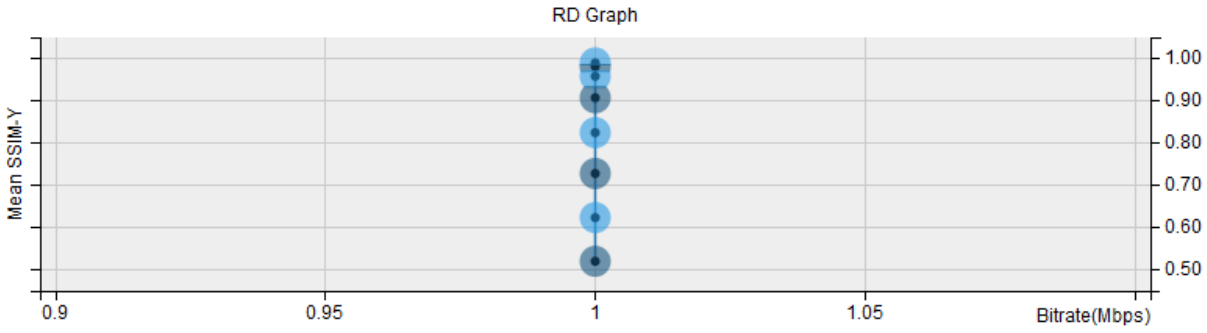

Figura 62. SSIM football_combt03×3 (Intel Video Quality Caliper).

Threshold15

Si se tienen en cuenta los resultados obtenidos para la métrica utilizada en este caso de estudio para el filtrado de mediana 3×3, era previsible que al aumentar el valor del threshold y aplicar la mediana a más píxeles, como consecuencia los valores de SSIM desciendan. Este decremento se puede apreciar en la Figura 63.

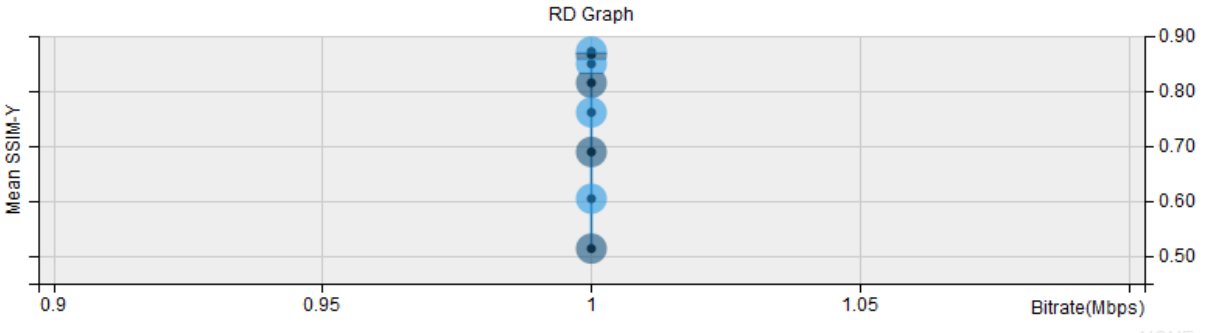

Figura 63. SSIM football_combt153×3 (Intel Video Quality Caliper).

Threshold30

Como se apuntaba en la sección anterior, el aumento de threshold trae consigo el descenso de los valores de SSIM (Figura 64), por lo que ésta es la peor de las tres combinaciones que utilizan la mediana 3×3 en lo que a la métrica de calidad se refiere.

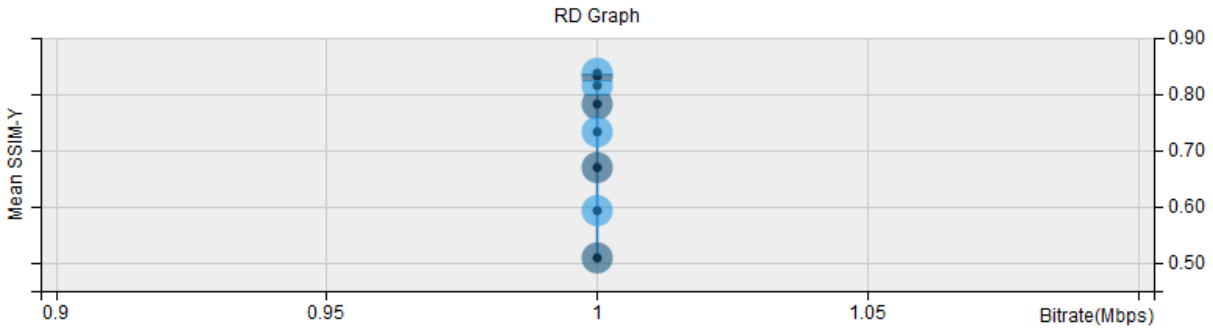

Figura 64. SSIM football_combt303×3 (Intel Video Quality Caliper).

4.2.2.5 Aplicación Combinado5×5 (híbrido)

Threshold30

Esta combinación es la que obtiene los valores de SSIM más altos de las tres que aplican el filtro de mediana 5×5. Dichos valores se pueden observar en la Figura 65.

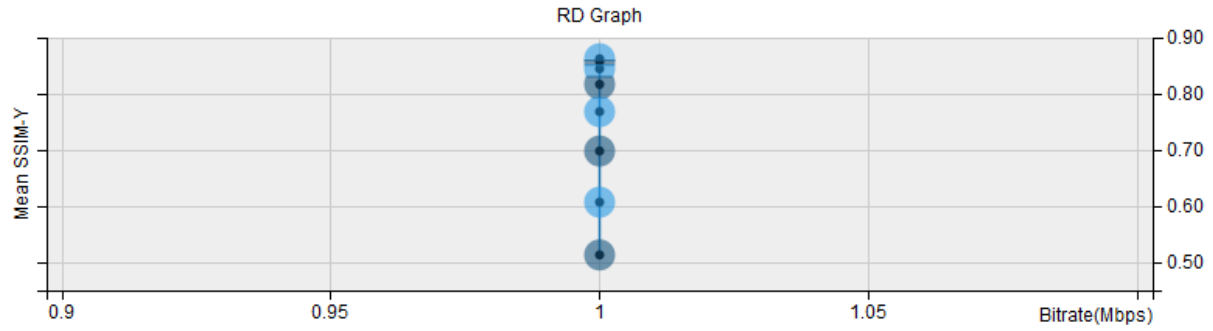

Figura 65. SSIM football_combt305×5 (Intel Video Quality Caliper).

Threshold40

Al aumentar el valor de threshold la mediana 5×5 filtra una mayor tasa de píxeles que incide en un empobrecimiento de los valores de SSIM obtenidos. Estos valores se pueden observar en la Figura 66.

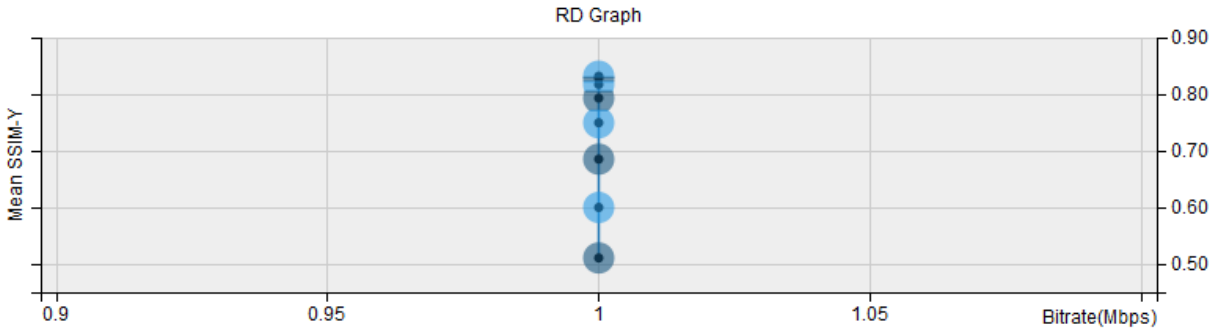

Figura 66. SSIM football_combt405×5 (Intel Video Quality Caliper).

Threshold60

Esta combinación es la peor de las tres en lo que tiene que ver con los valores de SSIM (al ser su threshold el que más se aproxima al ideal de los tres) y la mejor en lo que se refiere a tamaño de bitstream. Se muestran los valores de SSIM obtenidos para esta combinación en la Figura 67.

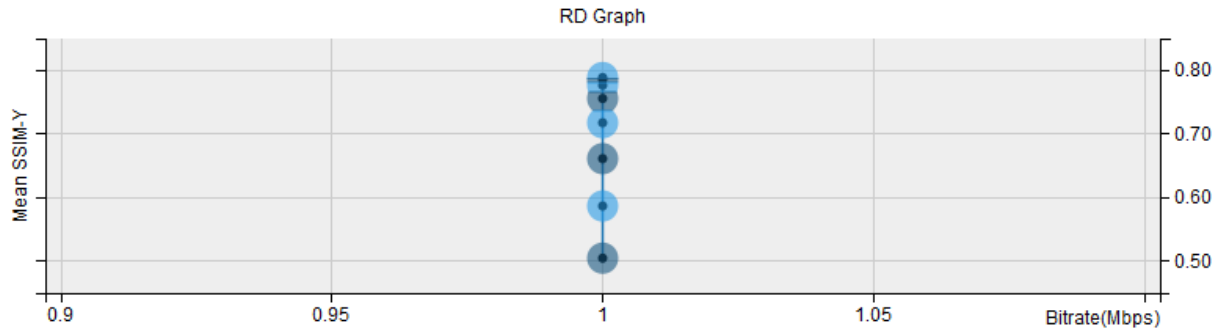

Figura 67. SSIM football_combt605×5 (Intel Video Quality Caliper).

4.2.3 Pareto experimento 3

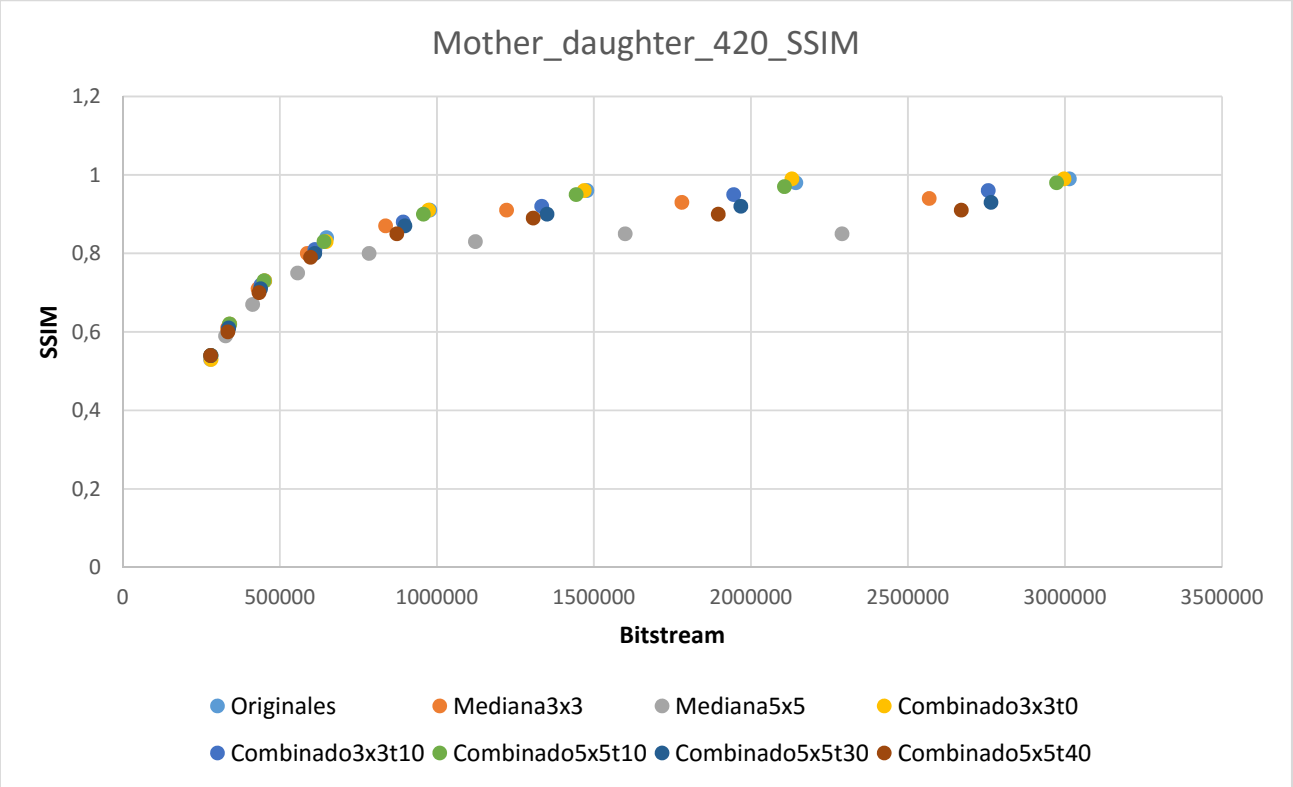

Figura 68. Experimento 3: Valores de SSIM para todas las codificaciones por categorías.

Los puntos que forman parte de la frontera de Pareto se dividen entre codificaciones con preprocesado en las que se utilizó sólo el filtro de mediana en sus dos versiones 3×3 y 5×5, codificaciones con preprocesado en las que se combinó el filtro de mediana con el operador de Sobel y codificaciones que se realizaron sobre el vídeo original. Dentro de las primeras existe una codificación con QP igual a 25 y en las segundas una con QP igual a 20, aunque de nuevo con un valor de threshold bajo.

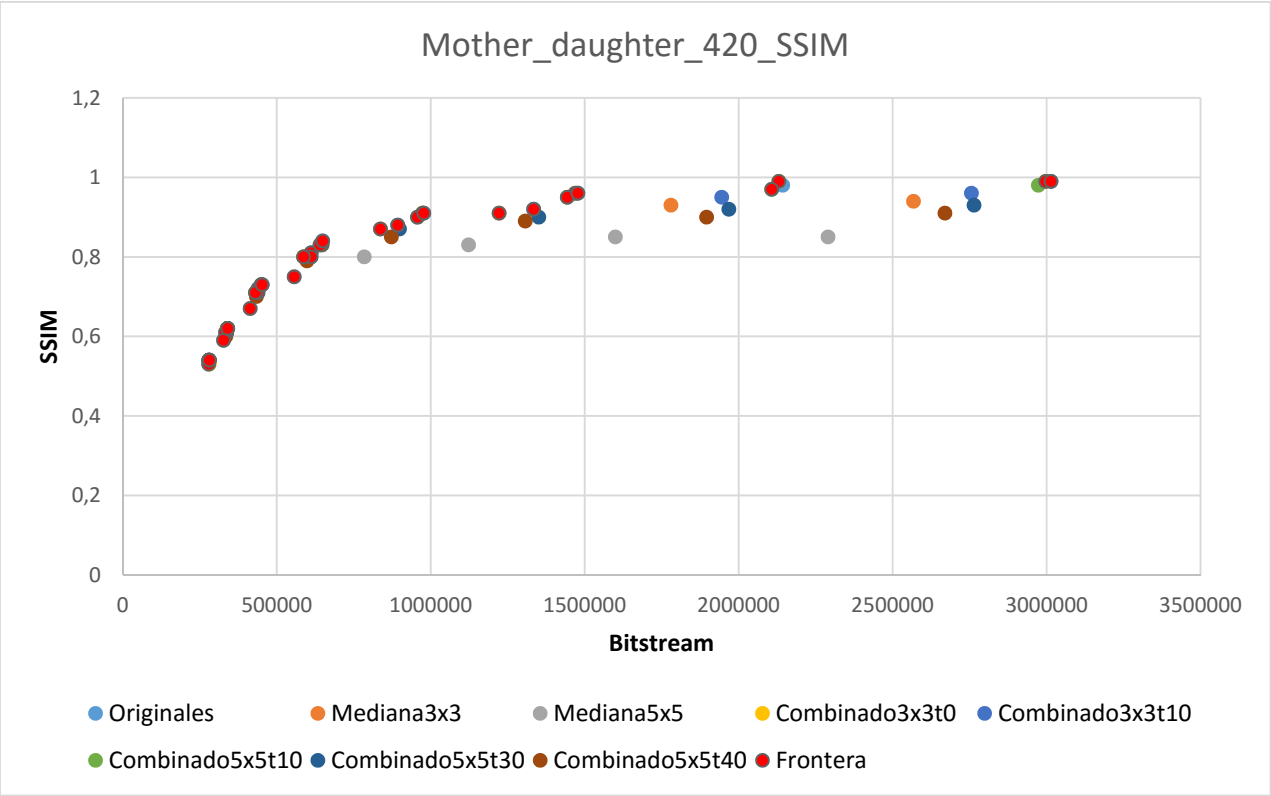

Figura 69. Experimento 3: Valores de SSIM para todas las codificaciones y frontera de Pareto.

4.2.4 Pareto experimento 4

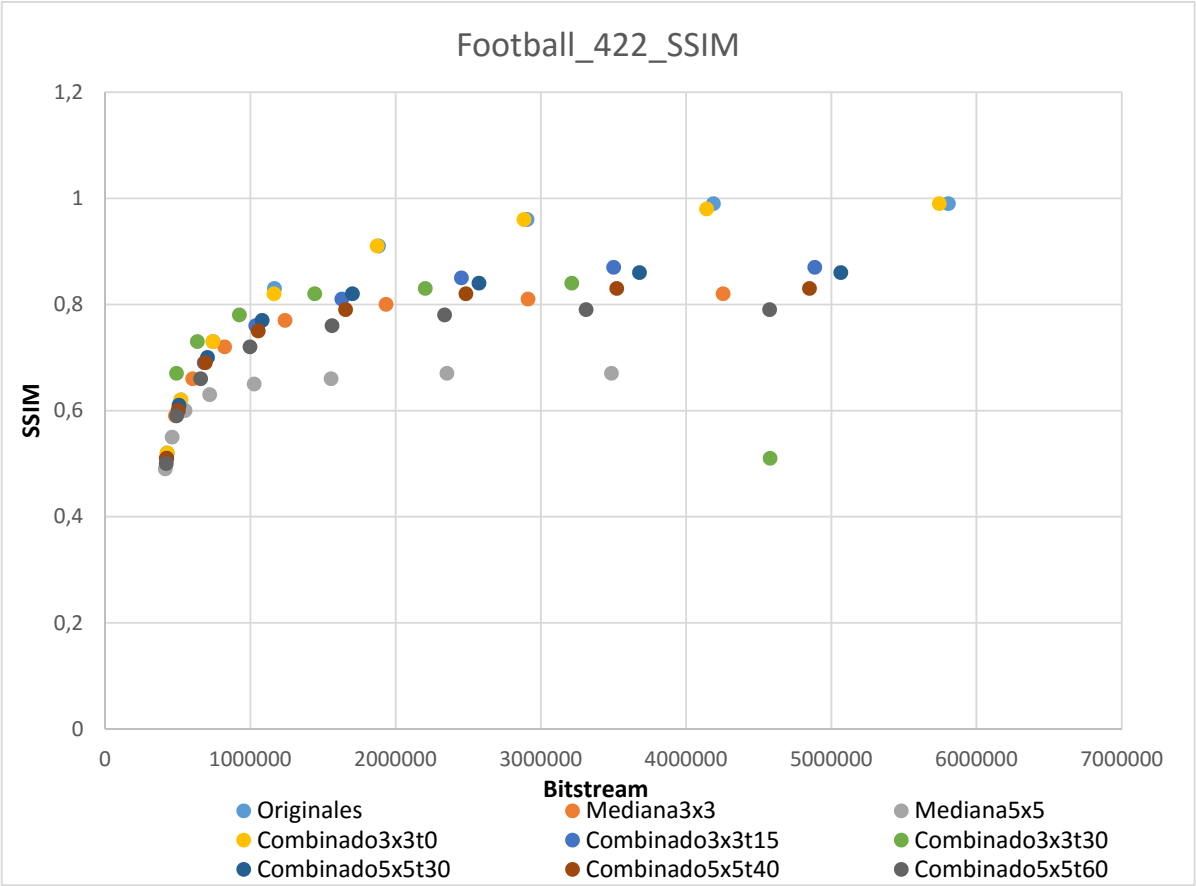

Figura 70. Experimento 4: Valores de SSIM para todas las codificaciones por categorías.

Como era previsible, se vuelven a tener las ocho codificaciones originales en la zona de dominancia, mientras que sólo cinco puntos de la frontera de Pareto tienen que ver con bitstreams obtenidos utilizando preprocesado con filtro de mediana. En las codificaciones restantes que aparecen en la frontera se ha utilizado preprocesado con combinación de filtro de mediana y operador de Sobel, en este caso existe una configuración con el mínimo QP aplicado (15), aunque también con el mínimo threshold (0). A diferencia de otros experimentos, existe dentro de estas últimas codificaciones una configuración con un threshold alto, resultando por otra parte el de máximo valor aplicado (60).

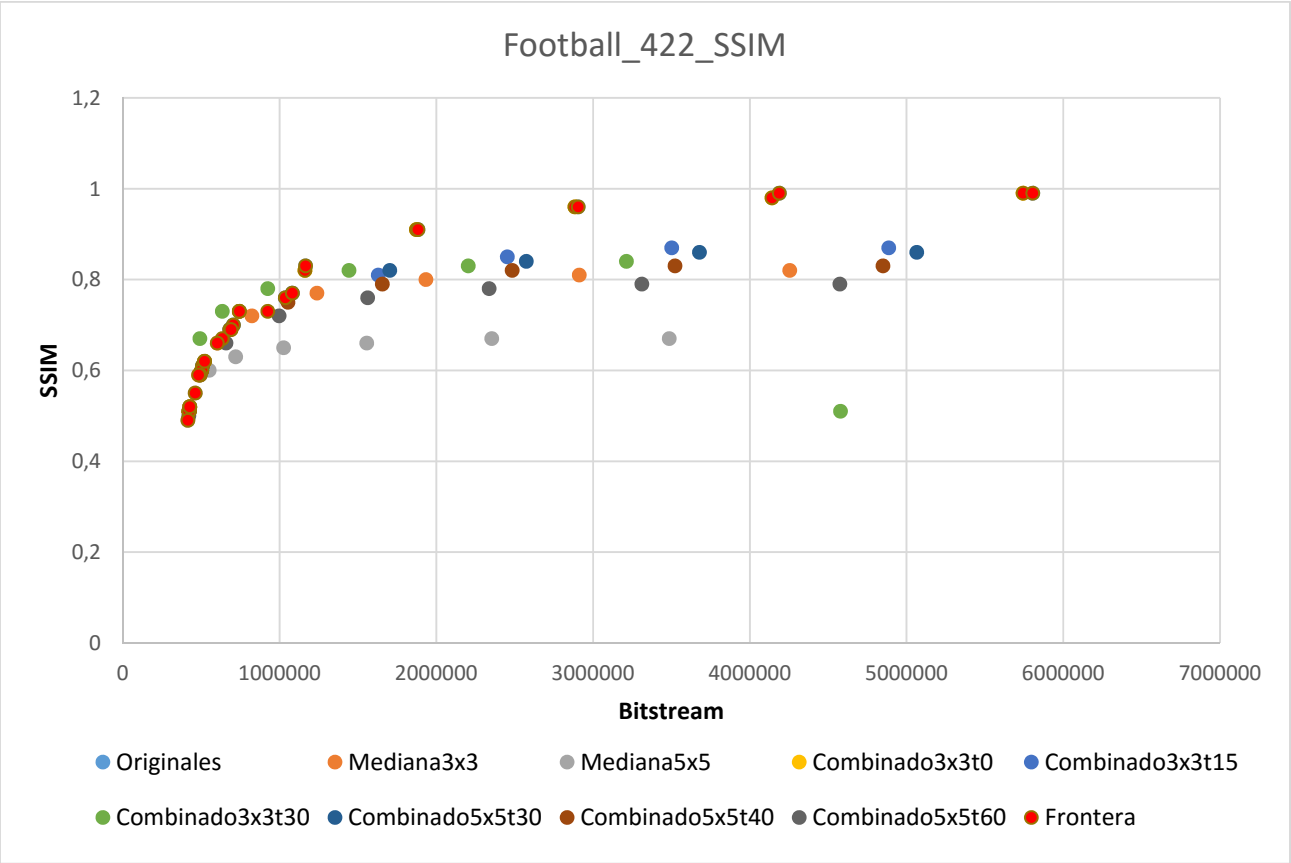

Figura 71 Experimento 4: Valores de SSIM para todas las codificaciones y frontera de Pareto.

Al igual que se hizo en los experimentos uno y dos, se muestran en la siguiente tabla resumen aquellas combinaciones con preprocesado que aparecen en la frontera de Pareto en los experimentos tres y cuatro. En este caso, los valores de thresholds más próximos al ideal y QP's más bajos muestran mejores resultados que los del primer caso de estudio (sobre todo en el experimento 3). Se reseñan en la Tabla 27 los más representativos.

Combinación	Bitstream	SSIM
Experimento 3		
Mediana 3×3, threshold = 10 y QP = 25	1334118	0,92
Mediana 5×5, threshold = 10 y QP = 25	1443736	0,95
Mediana 3×3	1221935	0,91
Experimento 4		
Mediana 3×3, threshold = 30 y QP = 35	924658	0,73
Mediana 5×5, threshold = 30 y QP = 35	1081475	0,77
Mediana 5×5, threshold = 60 y QP = 45	493321	0,59

Tabla 27. Resultados de las combinaciones más concluyentes de los experimentos tres y cuatro.

.

5. Conclusiones y trabajo futuro

Dentro del uso del codificador de vídeo de nueva generación H.265/HEVC, hay que resaltar que la inclusión de una etapa de preprocesado como la propuesta en este trabajo mejora la tasa de compresión del bitstream de salida, es decir se reduce su tamaño con respecto a las codificaciones que no utilizan la citada etapa de preprocesado. Si se necesita transmitir dicho bitstream, el canal de transmisión correspondiente necesitará un ancho de banda menor en caso de que se preprocese el vídeo con alguna de las alternativas planteadas anteriormente.

Por otra parte, en casi todos los casos observados se tiene una pérdida de calidad al utilizar preprocesamiento. Dicha pérdida es más flagrante cuando se observan los resultados obtenidos para la métrica PSNR. Esta métrica no está relacionada con la percepción que tiene el HVS de la imagen, por lo que sus resultados no son tan representativos del estudio que se pretende como los proporcionados por la métrica SSIM. Estos últimos son más esperanzadores e incluso muestran en algunos casos una reducción en tamaño de bitstream sin que ésta implique una pérdida de calidad.

Como trabajo futuro sería interesante utilizar métricas aún más relacionadas con la percepción que tiene de la imagen el HVS. El usuario final de los contenidos multimedia es el ser humano, por lo que la medida de la calidad debe enfocarse a su percepción. En este sentido existe una métrica denominada B-SSIM, desarrollada con este objetivo, que está más relacionada con la percepción humana que las anteriores.

Otra manera de plantear métricas bioinspiradas sería utilizar lo que se conoce como MOS (Mean Opinion Score), ésta es una métrica subjetiva que se basa en la visualización del vídeo decodificado por una muestra de observadores que puntúan la calidad del mismo utilizando un sistema llamado ACR en el que existen cinco posibles puntuaciones: excelente, buena, media, pobre y mala.

Como última propuesta de trabajo futuro, podría implementarse una etapa de postprocesado para incorporarla como última fase de la codificación, de manera que la Figura 22 se transformaría en la Figura 72 dando lugar a una realimentación del sistema global que ajuste cada uno de los parámetros de codificación de forma automática según condiciones del entorno.

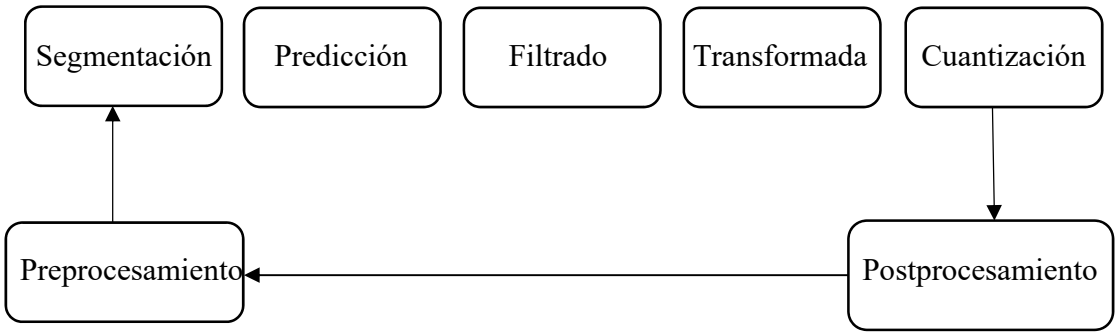

Figura 72. Fases de la codificación incluyendo preprocesamiento y postprocesamiento.

6. Bibliografía

1. ITU. https://www.itu.int/rec/T-REC-H.261-199303-I/es. [En línea]

2. x265.org/hevc-h265/. [En línea]

3. ITU. https://www.itu.int/rec/T-REC-H.263/es. [En línea]

4. PixelTools Corporation. http://www.pixeltools.com/rate_control_paper.html. [En línea]

5. Institute, Heinrich Hertz. https://hevc.hhi.fraunhofer.de. [En línea]

6. Intel. https://software.intel.com/en-us/qualify-for-free-software. [En línea]

7. D. G. Fernández ; A. A. Del Barrio ; Guillermo Botella ; Carlos García ; Uwe Meyer-Baese ; Anke Meyer-Baese; HEVC optimizations for medical environments. Proc. SPIE 9871, Sensing and Analysis Technologies for Biomedical and Cognitive Applications 2016, 98710B (May 19, 2016);

8. Richardson, Iain. https://www.vcodex.com/h264avc-intra-precition/. [En línea]

9. Richardson, Iain E. G. H.264 and MPEG-4 Video: Video Coding for Next-generation Multimedia. s.l. : John Wiley & Sons.

10. Wiegand, Thomas. https://www.hhi.fraunhofer.de/en/departments/vca/research-groups/image-video-coding/research-topics/h264mpeg4-avc/tree-structured-partitioning-for-motion-compensated-prediction.html. [En línea]

11. D. Guillermo Fernandez, Alberto A. del Barrio, Guillermo Botella, Carlos García, "4K-based intra and interprediction techniques for HEVC", Proceedings of SPIE Vol. 9897, 98970B (2016) SPIE Digital Library

12. Ramírez, Gustavo. http://www.enelinternetdelascosas.org/2009/08/que-es-un-filtro-de-respuesta-finita.html. [En línea]

13. Weeks, Michael. *Digital Signal Processing Using Matlab And Wavelets.* s.l. : Jones & Bartlett Learning, 2006.

14. Taylor, Stewart. http://www.drdobbs.com/h264-and-video-compression/201203492. [En línea]

15. Martinsanz, Gonzalo Pajares. *Vision por computador: imágenes digitales y aplicaciones.* 2ª. s.l. : RA-MA S.A. Editorial y Publicaciones , 2007.

16. Richardson, Iain. https://www.vcodex.com/h264avc-4×4-transform-and-quantization/. [En línea]

17. Malepati, Hazarathaiah. *Digital Media Processing: DSP Algorithms Using C.* s.l. : Newnes.

18. Ozer, Jan. http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-Is-HEVC-(H.265)-87765.aspx. [En línea]

19. Dissanayake, M. B., & Abeyrathna, D. L. (2015). Performance comparison of HEVC and H. 264/AVC standards in broadcasting environments. *J Inf Process Syst*, *11*(3), 483-494.

20. http://forum.doom9.org/showthread.php?t=167081. [En línea]

21. videoultraaltadefinicao. https://videoultraaltadefinicao.wordpress.com/o-codec-hevch-265/. [En línea]

22. Sze , Vivienne y Budagavi, Madhukar. http://www.rle.mit.edu/eems/wp-content/uploads/2014/06/H.265-HEVC-Tutorial-2014-ISCAS.pdf. [En línea]

23. Santamaria, Pedro. http://www.xatakahome.com/televisores/4k-o-uhd-una-cuestion-de-marketing. [En línea]

24. Narang, Nitin. http://www.mediaentertainmentinfo.com/2013/10/4-concept-series-what-is-the-difference-between-hevc-h-265-and-h-264-mpeg-4-avc.html/. [En línea]

25. https://www.videomaker.com/videonews/2012/10/the-difference-between-raw-video-and-uncompressed-video. [En línea]

26. http://arte.about.com/od/Diccionario-De-Arte/ss/Que-es-color.htm#step2. [En línea]

27. http://arte.about.com/od/Diccionario-De-Arte/ss/Que-es-color.htm#step3. [En línea]

28. elektronika-dasar.web. http://elektronika-dasar.web.id/wp-content/uploads/2012/11/. [En línea]

29. www.efectohd.com/2008/09/entender-la-correcion-de-gamma.html. [En línea]

30. Intel. https://software.intel.com/en-us/node/503874. [En línea]

31. Microsoft. https://msdn.microsoft.com/es-es/library/windows/desktop/bb530104(v=vs.85).aspx. [En línea]

32. Intel. https://software.intel.com/en-us/node/503876. [En línea]

33. crazybiocomputing. http://crazybiocomputing.blogspot.com.es/2012/09/rgb-file-packed-or-planar.html. *http://crazybiocomputing.blogspot.com.es/2012/09/rgb-file-packed-or-planar.html.* [En línea]

34. http://forum.doom9.org/showthread.php?t=168947. [En línea]

35. Grange, Adrian, de Rivaz, Peter y Hunt, Jonathan. *VP9 Bitstream & Decoding Process Specification.*

36. http://alojamientos.us.es/gtocoma/pid/tema3-1.pdf. [En línea]

37. http://www.rinconastur.com/php/php134.php. [En línea]

38. Claveria, Alberto. http://www.albertoclaveriafoto.com.ar/blog/?p=298. [En línea]

39. Martínez, Silvia Satorres. http://www4.ujaen.es/~satorres/practicas/practica3_vc.pdf. [En línea]

40. *Image Quality Assessment: From Error Visibility to Structural Similarity.* Wang, Zhou, y otros. 4, s.l. : IEEE Transactions on image processing, 2004, Vol. 13.

41. *Técnicas de evaluación de la calidad de la imagen. Tendencias y métricas basadas en bordes.* Silvestre Blanes, Javier y Gorricho, Juan Luis. Valencia : s.n.

42. Wang, Z., Bovik, A. C., & Sheikh, H. R. (2005). Structural similarity based image quality assessment. *Digital Video image quality and perceptual coding*, 225-241.

43. Basavaraju, S., & Sivakumar, B. (2015, February). Modified pre and post processing methods for optimizing and improving the quality of VP8 video codec. In *Electronics and Communication Systems (ICECS), 2015 2nd International Conference on* (pp. 1346-1354). IEEE.

44. http://www.cns.nyu.edu/pub/lcv/wang03-reprint.pdf. [En línea]

45. forum.doom9.org. http://forum.doom9.org/showthread.php?t=168947. [En línea]